

ESPECIFICAÇÃO DE TRABALHO PRÁTICO

Análise e Desenvolvimento de Sistemas
INF008 – Programação Orientada a Objetos
Professor: Sandro Santos Andrade

1. Objetivo Geral

Desenvolver um sistema orientado a objetos, em Java, para gerenciar eventos acadêmicos em uma universidade, com foco em aplicar os principais conceitos de orientação a objetos, como herança, interfaces, sobreposição, polimorfismo, collections e organização modular.

2. Descrição do Sistema

A universidade promove diversos eventos acadêmicos, que podem incluir: palestras, workshops, cursos de curta duração e feiras acadêmicas. Cada tipo de evento possui características específicas, mas todos compartilham elementos em comum, como: título, data, local, capacidade de participantes e descrição.

3. Requisitos Funcionais

O sistema deve permitir:

1. Cadastro de eventos de diferentes tipos (palestras, cursos, workshops, feiras).
2. Associação de participantes a eventos (com controle de vagas disponíveis).
3. Geração de certificados (texto) com dados do evento e do participante.
4. Participantes podem ser alunos, professores ou externos, com dados específicos para cada categoria. Os participantes dos cursos devem ser exclusivamente alunos (não podem ser professores ou externos).
5. Suporte a eventos híbridos (presenciais ou online), com comportamentos diferentes para o processo de inscrição.
6. Relatório de eventos por tipo e data.

4. Requisitos Técnicos

O sistema deve obrigatoriamente contemplar os seguintes tópicos:

- Conceitos Básicos
 - Classes, objetos e referências
 - Atributos, métodos e visibilidade (*private*, *protected*, *public*)
 - Pacotes organizados logicamente
 - Arquivos JAR e uso de *classpath*
 - Métodos construtores
- Herança e Polimorfismo
 - Criar uma hierarquia de classes para eventos e outras entidades
 - Utilizar interfaces ou classes abstratas para definir comportamentos genéricos
 - Demonstrar uso ligação dinâmica
- Collections
 - Utilizar a collection mais adequada para cada situação
 - Demonstrar *casting* seguro e uso correto das hierarquias de collections
- Outros Requisitos
 - Atributos e métodos estáticos
 - Sobrecarga de métodos
 - Projeto estruturado com uso de pelo menos uma biblioteca externa
 - Interface de linha de comando (não é necessária interface gráfica)



Boas Práticas:

- Use nomes significativos para variáveis, métodos e classes.
- Divida a lógica em métodos pequenos e reutilizáveis.
- Siga as convenções do programação do próprio Java.
- Lembre-se que uma solução simples é sempre melhor.

5. Critérios de Avaliação

Critério	Pontos
Modelagem OO adequada (herança, polimorfismo, interfaces)	2,0
Uso correto de collections, casting e hierarquias	1,5
Organização do projeto, pacotes e estrutura Maven	1,0
Aplicação de métodos estáticos e sobrecarga	1,0
Visibilidade e uso de construtores	1,0
Aplicação prática de ligação dinâmica	1,0
Geração de JAR e execução correta	1,0
Clareza, legibilidade e boas práticas no código	1,5

6. Entregáveis

- Código-fonte do sistema + arquivo README.md compactados em formato *.zip* ou *.tar.gz*.
 - **OBS:** o pacote *.zip* ou *.tar.gz* não deve conter arquivos *.class*, apenas arquivos *.java*. Limpe o seu projeto antes de criar o pacote.
- Arquivo README.md deve conter:
 - Instruções de compilação e execução.
 - **OBS:** a compilação e execução do projeto não deve requerer a instalação de nenhuma IDE. Se você não quiser utilizar nenhuma *build tool*, como o *maven* ou *gradle*, informe como compilar e executar os arquivos manualmente com os comandos *javac* e *java*. Caso utilize alguma *build tool*, informe os comandos necessário para compilar e executar sua aplicação.

7. Prazo e Forma de Entrega

- O trabalho deve ser enviado até o dia **11/06/2024** às 23:59:59 (não extensível).
- O código-fonte e o arquivo README.md deve ser enviado por email para sandroandrade@ifba.edu.br, com o assunto "INF008 T1 <seu-nome-completo>", sem as aspas e sem os caracteres < e >. Ex: INF008 T1 Eduardo da Silva Santos.
- Trabalhos enviados com outro assunto não serão corrigidos.

8. Informações Importantes

- Todos os códigos-fonte entregues serão checados por plágio utilizando as ferramentas **MOSS** (Measure of Software Similarity - <https://theory.stanford.edu/~aiken/moss/>) e **JPlag** (<https://github.com/jplag/JPlag>). Desenvolva sua própria solução, é sua chance de aprender.
- Todas as dúvidas sobre o trabalho deverão ser abertamente discutidas no grupo da disciplina no Telegram.

Bom trabalho!