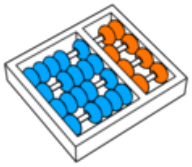

Universidade Estadual de Campinas
Instituto de Computação

Prof. Dr. Bruno Barbieri de Pontes Cafeo
cafeo@ic.unicamp.br
https://ic.unicamp.br/~cafeo/

INF1900 - Recursos Avançados de C++ (2023)
Exercício 3 - Programação Genérica e Templates, Semântica de Transferência, Smart Pointers

Contexto

Imagine que você está trabalhando em um sistema de gerenciamento de arquivos e precisa implementar uma classe `FileResource` que representa um recurso de arquivo. A classe `FileResource` deve conter um ponteiro inteligente para gerenciar o recurso de arquivo. Além disso, você deseja que a classe seja genérica o suficiente para lidar com diferentes tipos de arquivos, como arquivos de texto, imagens e áudio.

Parte 1 - Programação Genérica e Templates

Implemente a classe `FileResource` como uma classe de modelo (`template`) que pode ser parametrizada com o tipo de dado do arquivo. A classe `FileResource` deve incluir os seguintes métodos:

- `FileResource::FileResource(const std::string& filename)`: Um construtor que recebe o nome do arquivo e abre o arquivo correspondente.
- `FileResource::ReadData`: Um método que lê e retorna os dados do arquivo.
- `FileResource::WriteData`: Um método que escreve os dados do tipo `T` para o arquivo recebidos via argumento.

Certifique-se de que sua implementação seja genérica o suficiente para lidar com diferentes tipos de arquivos, como texto, imagens e áudio.

Parte 2 - Semântica de Transferência

Você percebe que é importante fornecer suporte para a semântica de transferência (move semantics) ao trabalhar com objetos `FileResource`. Portanto, implemente o seguinte método:

- `FileResource::MoveTo`: Um método que move o recurso de arquivo de um objeto `FileResource` para outro. Certifique-se de que, após a transferência, o recurso seja gerenciado corretamente.

Parte 3 - Smart Pointers

Para garantir que os recursos de arquivo sejam liberados adequadamente, você decide utilizar ponteiros inteligentes. Altere sua classe `FileResource` para usar um `std::unique_ptr` para gerenciar o recurso de arquivo. Certifique-se de implementar adequadamente a liberação de recursos quando o objeto `FileResource` é destruído.