Universidade Federal de Ouro Preto - UFOP
Instituto de Ciências Exatas e Biológicas - ICEB
Departamento de Computação - DECOM
Disciplina: Programação Orientada a Objetos
Professor: Guillermo Cámara Chávez

TRABALHO PRÁTICO I

Instruções:

- O problema deve ser resolvido por meio de um programa em C++ e a STL;
- Utilize o site http://www.cplusplus.com/ como referência;
- O código-fonte deve estar devidamente comentado;
- Não serão aceitos trabalhos que caracterizem cópia (mesma estrutura e algumas pequenas modificações) de outro ou de códigos da internet;
- Eventualmente, após a entrega dos trabalhos serão marcadas entrevistas com cada um dos alunos para apresentação dos mesmos para o professor.

Entrega:

- A entrega do código-fonte será feita pelo *Moodle*até o dia **15 de Setembro**.
- Deve ser entregue um zip com:
  - Relatório (descrito ao fim deste texto);
  - Código fonte;
  - Descrição de como compilar e executar o projeto pela linha de comando.
  - Vídeo mostrando a compilação, execução e uso do código

Avaliação:

- Funcionamento adequado do programa
  - Códigos que não compilarem e tiverem *warnings* diminuirão a nota;
  - Corretude (independente se gerado por IDE ou manualmente).
- Atendimento ao enunciado do trabalho;
- Comentários;
- Identação do código e boas práticas de programação;
- Boa organização do código fonte em geral;
- Bom uso da STL.

Enunciado

O objetivo deste trabalho é a implementação de um sistema de gerenciamento de uma coleção de livros. Essa coleção de livros deverá ser implementada baseada na *Standard Templates Library* (STL).

A escolha dos contêineres da STL devem ser adequados ao contexto da sua utilização, isto é, a sua escolha deve ser pensada de acordo com suas funcionalidade e os algoritmos de manipulação. O objetivo não é usar o contêiner somente como um armazenador de dados, mas sim utilizar suas funções. Além disso, deve-se priorizar sempre os algoritmos que já estão implementados na STL ao invés de reimplementar alguma função já existente.

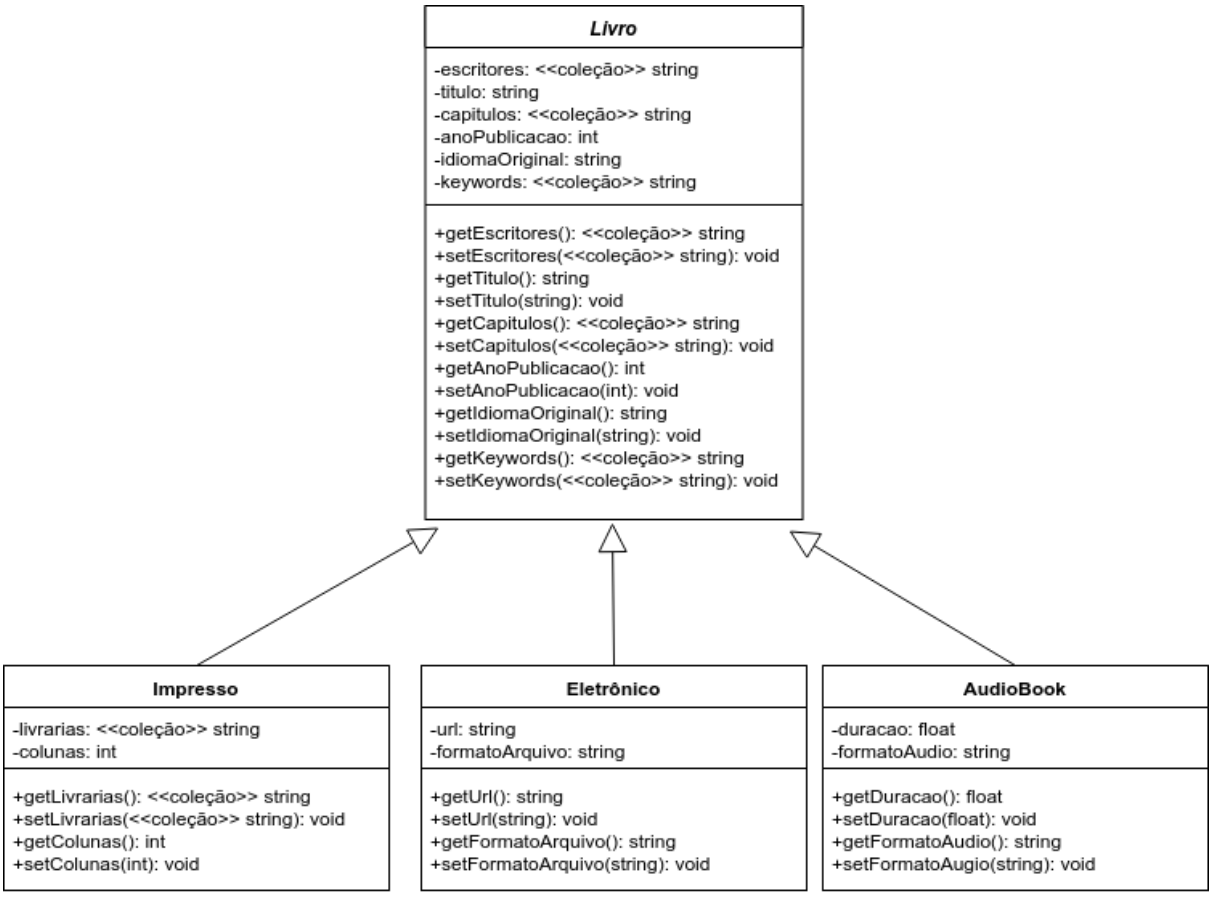
São enviados 16 arquivos cada um com dados de um único livro. Assim que o sistema for iniciado, eles devem ser lidos para preencher a coleção. O padrão do arquivo é:

- Linha 1 - tipo do livro (1 - Impresso; 2 - Eletrônico e 3 - Audiobook);
- Linha 2 - Nome do livro;
- Linha 3 - Escritores separados por ponto e vírgula (;);
- LInha 4 - Ano de publicação;
- Linha 5 - Idioma de publicação;
- Linha 6 - *Keywords* separadas por ponto e vírgula (;);
- Linha 7 - Capítulos do livro separados por ponto e vírgula (;);
- Linha 8/9 - Depende do tipo de livro:
    - Impresso:
        - Livrarias onde podem ser encontrados os livros;
        - Número de colunas do arquivo impresso.
    - Eletrônico:
        - URL de onde pode ser comprado;
        - Formato do arquivo.
    - Audiobook:
        - Duração do áudio;
        - Formato do áudio.

Ao final deste arquivo há uma função. Ela pode ser usada para quebrar as linhas que utilizam ponto e vírgula para separar os escritores, *keywords* e capítulos em coleções. Altere a palavra *Container* para o tipo de coleção que você desejar.

Detalhes do sistema

O sistema conta com três tipos de livros cadastrados: livro impresso, livro eletrônico e audiobook. O diagrama UML abaixo indica quais os atributos e métodos dos tipos de livros cadastrados. No diagrama, o termo *<<coleção>>* indica que algum contêiner da STL deverá ser usado.



O sistema em si será codificado no *main* do programa. Todas as operações descritas devem ser implementadas em forma de função e chamadas no *main*. O *main* deverá conter uma única coleção de objetos de livros impressos, eletrônicos e audiobooks. A coleção criada deve possibilitar a manipulação e posteriormente transformações para as seguintes operações:

a. Sobrecarregar o operador **<<** (saída) para todas as classes. Todas as operações que pedirem somente para exibir o livro devem usar esse operador. Quando for exibir as coleções com **todos os livros (impresso, eletrônico e audiobook)** deixar de lado as características específicas, exceto quando solicitado. Entre a impressão de cada parâmetro especificado deve-se usar o caractere '|'. A impressão deve ser feita de forma estruturada seguindo o seguinte padrão:
    i. Mostrar o título com 30 caracteres (caso tenha mais caracteres, cortar a string para que ela fique com 30). Justificar à esquerda;
    ii. Mostrar o nome do primeiro escritor com 30 caracteres (caso tenha mais caracteres, cortar a string para que ela fique com 30). Justificar à direita;
    iii. Mostrar o idioma com 10 caracteres (caso tenha mais caracteres, cortar a string para que ela fique com 10). Justificar a esquerda;
    iv. Mostrar o número de capítulos com 3 caracteres. Se caso o número de capítulos for menor que o número 10, completar os dois caracteres

com espaços. Menor que o número 100, completar os caracteres com dois espaços.

v. Mostrar o número de keywords com 2 caracteres. Se caso o número de capítulos for menor que o número 10, completar o carácter das casas da dezena com zero.

vi. Mostrar uma característica específica do tipo de publicação com 10 caracteres:

1. Impresso: Primeira livraria se houver, senão, “Nenhuma”;
2. Eletrônico: formato do arquivo;
3. AudioBook: duração com duas casas decimais.

Título	Primeiro escritor	Idioma	N° de capítulos	N° de *Keywords*	Característica específica

b. Dado um idioma, criar uma função que retorne uma **única** coleção com **todos os livros (impresso, eletrônico e audiobook)** e no *main* exibir o nome de todos os livros com o Idioma especificado.

c. Dado um formato de livro, criar uma função que retorne uma coleção de **livros eletrônico** ordenada pelo ano de publicação e no *main* exibir os livros retornados.

d. Dado um número de livrarias, criar uma função que retorne uma coleção com os **livros impressos** com um número maior ou igual de livros em livrarias. No main deve ser mostrado uma mensagem de “não encontrado” se **nenhum livro impresso** for encontrado e, caso contrário, exibir o nome do livro e a quantidade de livrarias.

e. Dado um escritor, criar uma função que retorne *true* se houver algum **audiobook** desse escritor, ou *false* se caso não houver. No *main* deve ser exibido se há ou não.

f. Dado o título de um livro, criar uma função que retorne uma coleção com **todos os livros (impresso, eletrônico e audiobook)** com esse título e no *main* exibir todos os dados dos livros que forem retornados.

g. Criar uma função que retorne uma coleção com todos os *keywords* disponíveis. Não deve haver repetição. Exibir no *main* os *keywords*.

h. Dado um número de capítulos, criar uma função que retorne uma coleção com **todos os livros (impresso, eletrônico e audiobook)** com esse número ou menos de capítulos e ordenada pelo nome do primeiro escritor e no *main* exibir os livros.

i. Dado o nome de um livro, criar uma função que “retorna” um *iterador* para **cada tipo de livro** apontando para o elemento encontrado. No *main* deve ser mostrado se o livro foi encontrado e os dados do mesmo para cada um dos tipos de livro (mostrar os dados comuns e específicos do livro conforme o item a).

j. Criar uma função que recebe uma **única** coleção de **livros de todos os tipos** e que mostre no terminal ou salve em um arquivo (saida.txt) **todos os tipos de livros**. Mostrar os dados comuns e específicos do livro conforme o item a). Um argumento

passado para a função define qual será a saída. Nesse caso é necessário *downcasting*.

k. Dado uma *keyword*, criar uma função que retorne a quantidade de livros que possuam aquela *keyword* dentro uma coleção, seja ela qual for (livro, impresso, eletrônico ou audiobook). No *main* deve ser mostrado a quantidade encontrada.

l. Criar uma função que recebe uma **única** coleção de **livros de todos os tipos**, além de uma estrutura capaz de realizar o mapeamento a seguir (No *main* mostrar o resultado):

    i. Inglês = ING;
    ii. Espanhol = ESP;
    iii. Francês = FRS;
    iv. Português = POT.

Relatório

Deverá ser entregue um relatório descrevendo o porquê da escolha de cada contêiner e algoritmo em cada parte da implementação. Além disso, dado os contêineres presentes na STL, deve ser descrito **sucintamente** qual o melhor contêiner para cada uma das seguintes situações:

- Acessar uma posição específica de um contêiner;
- Adicionar um elemento e manter somente elementos únicos no contêiner;
- Inserção no final;
- Remoção no final;
- Retornar um valor baseado em uma chave específica (não necessariamente inteiros);
- Inserção no início;
- Remoção no início;
- Busca por um elemento;
- Contêiner com o comportamento de primeiro a entrar é o último a sair;
- Contêiner com o comportamento de primeiro a entrar é o primeiro a sair.

Função auxiliar

```
#include <string>
#include <sstream>
#include <algorithm>
#include <iterator>
using namespace std;
void split(const string& str, Container& cont, char delim = ' ') {
    stringstream ss(str);
    string token;
    while (getline(ss, token, delim))
        cont.push_back(token);
}
```