INF1007 - PROGRAMAÇÃO II

LISTA DE EXERCÍCIOS 11

1. Um número racional é um número que corresponde a uma razão (ou fração) de dois valores inteiros: um numerador e um denominador. Supondo que um número racional seja representado pela seguinte estrutura:

```
struct racional
{
  int numerador;
  int denominador;
};
```

Implemente o TAD Racional (interface `racional.h` e implementação `racional.c`) para representar e realizar operações aritméticas em números racionais representados pela estrutura `racional`.

O TAD Racional deve exportar e implementar as seguintes funções:

- `racional_cria`: a função recebe dois números inteiros representado os valores do numerador e denominador. A função deve criar e retornar um ponteiro para a representação racional destes números;

  ATENÇÃO: a representação interna do racional deve corresponder à forma reduzida da fração correspondente. Por exemplo, se o usuário entrar "4/6", deve-se armazenar "2/3".

  Dica: use a função mdc para fazer a transformação necessária:

```
int mdc(int x, int y)
{
  if (y == 0)
    return x;
  return mdc(y, x % y);
}
```

- `racional_mostra`: a função recebe um ponteiro para um valor representado pelo tipo abstrato Racional e exibir na tela o numerador e o denominador, como uma fração, por exemplo 2/3 (isto é, colocando uma '/' entre eles);

- `racional_libera`: a função recebe um ponteiro para um número Racional e libera a memória utilizada pelo número racional;

- `racional_soma`: a função recebe dois ponteiros para o tipo abstrato Racional e retorna um ponteiro para um Racional contendo a soma dos números racionais recebidos por parâmetro. Lembre-se de reduzir o resultado com a função mdc;

- `racional_multiplica`: a função recebe dois ponteiros para o tipo abstrato Racional e retorna um ponteiro para um Racional contendo o produto dos números racionais recebidos por parâmetro. Lembre-se de reduzir o resultado com a função mdc;

- `racional_compara`: a função recebe dois ponteiros para o tipo abstrato Racional e retorna um valor menor que zero se o primeiro Racional é menor que o segundo, zero se são iguais e um valor maior que zero se o primeiro Racional é maior que o segundo.

Utilizando o TAD Racional, crie o módulo principal de um programa (calculadora.c) que implemente uma calculadora baseada em números racionais. A calculadora deve permitir que o usuário realize as operações de soma, multiplicação e comparação de números racionais, exibindo os resultados das operações na tela. Obrigatoriamente todos os números utilizados pela calculadora devem ser representados pelo tipo abstrato Racional.

2. A sua equipe de programadores está trabalhando no desenvolvimento de um sistema para o gerenciamento de livros de uma biblioteca. A sua tarefa é implementar um TAD para representar os livros neste sistema.

   Sabe-se que um livro é representado pelo seguinte tipo estruturado:

```
struct livro
{
  char titulo[50];
  char autor[30];
  char genero[10];
  int ano;
};
```

   As funções que devem ser exportadas pelo TAD Livro (na interface do TAD), são as seguintes:

   - A função `livro_cria` que recebe por parâmetro o título, autor, gênero e ano de publicação do livro, cria um livro com esses dados e retorna um ponteiro para o novo Livro.

   - Quatro funções de obtenção dos dados armazenados em um TAD Livro (denominadas `livro_obtemGenero`, `livro_obtemAutor`, `livro_obtemTitulo`, e `livro_obtemAno`) que recebem um ponteiro para Livro e retornam o valor em questão (e.g. `livro_obtemTitulo` retorna (o ponteiro

para) a cadeia de caracteres do titulo de um TAD Livro recebido, assim como `livro_obtemAno` retorna o ano de um TAD Livro recebido).

- Uma sexta função, denominada `livro_verificaNoModernismo` que recebe um ponteiro para Livro e verifica se esse livro pertence ao segundo período do modernismo brasileiro (1930 a 1945). Esta função retorna -1 se o ano da obra for anterior a 1930, retorna 0 se for no período 1930 a 1945, e retorna 1 se o ano for posterior a 1945.

Escreva o conteúdo do arquivo `livro.h` com a interface deste TAD Livro, incluindo a definição da estrutura e também todas as funções disponíveis para o usuário. Em seguida, escrevendo o módulo (arquivo `livro.c`) que implementa todas as funções do TAD Livro.

Em seguida, escreve o modulo principal de um programa (`principal.c`) que crie e inicialize um vetor de ponteiros para Livro utilizando as funções disponibilizadas pelo TAD e inserindo neste vetor os seguintes livros:

```
"Novos Poemas","Vinicius de Morais","poesia",1938
"Poemas Escritos na India","Cecilia Meireles","poesia",1962
"Orfeu da Conceição","Vinicius de Morais","teatro",1954
"Ariana, a Mulher","Vinicius de Morais","poesia",1936
```

No modulo principal do seu programa, implemente as seguintes funções:

- Usando o TAD Livro, escreva a função `ordenaLivros` que ordena o vetor de ponteiros para Livros em ordem crescente por nome de autor, depois por gênero e finalmente por ano de publicação. A função recebe um vetor de ponteiros para Livro e o número de livros existentes no vetor.

- Usando o TAD Livro, escreva a função de busca binária `buscaLivroModerno` que recebe um vetor de ponteiros para Livro, o número de livros, um nome de autor e um gênero, e retorna um ponteiro para o livro mais recente do autor, do tipo indicado, no segundo período do modernismo brasileiro. Considere que o vetor está ordenado conforme o critério utilizado pela sua função `ordenaLivros`.

  Por exemplo, para o vetor de ponteiros criado no programa principal, se a função receber os argumentos autor="Vinicius de Morais" e gênero="poesia", ela deve retornar um ponteiro para: "Novos Poemas","Vinicius de Morais","poesia",1938. Caso nenhum livro atendendo à descrição seja encontrada, a função retorna NULL.

Em seguida, continue a implementação do modulo principal (`principal.c`) utilizando as funções criadas para ordenar o vetor de ponteiros para Livro e em seguida buscar um livro no vetor. O seu programa deve exibir na tela o vetor ordenado e também o livro retornado pela busca.