Modalidade: em grupo (no máximo 3 alunos por grupo)
Data de entrega: 24/maio até 23:59 na tarefa do Microsoft Teams. Entregar UM ÚNICO ARQUIVO COMPACTADO .ZIP CONTENDO APENAS OS ARQUIVOS FONTES, e um arquivo *readme* no formato TXT, informando os nomes dos alunos do grupo e instruções para uso do programa. São descontados 25% da nota por dia de atraso.

Fazer um programa em linguagem C para gerenciar o cadastro de produtos de um *site* de vendas online, utilizando árvore B* implementada em **arquivo binário**, conforme descrito a seguir:

- existem vários produtos cadastrados, contendo as seguintes informações:
  - código: número inteiro que identifica univocamente cada produto
  - nome: nome do produto, tendo no máximo 50 caracteres
  - marca: marca do produto, tendo no máximo 30 caracteres
  - categoria: descreve a categoria do produto, tendo no máximo 50 caracteres
  - estoque: número de unidades disponível no estoque
  - preço: preço unitário do produto.
- O sistema deve ter as seguintes funcionalidades implementadas, organizadas convenientemente em menus:
  - cadastrar produto: cadastra um produto a partir das informações fornecidas pelo usuário
  - remover produto: remove um produto do cadastro, a partir do seu código
  - atualizar preço: atualiza o preço de um produto, a partir de seu código
  - atualizar estoque: atualiza o estoque de um produto, a partir de seu código
  - imprimir informações de um produto, a partir de seu código
  - imprimir lista de todos os produtos: utilizando varredura *in-ordem*, listando os códigos e respectivos nomes dos produtos
  - imprimir árvore: imprime apenas os códigos dos produtos formatados visualmente como uma árvore B*. Considerando a árvore da Figura 1, a saída deve ser:

```
[8,22,40]
[2,3,4,5,6] [9,12,15,20] [25,27,30,31] [43,48,55,67,70,72]
```

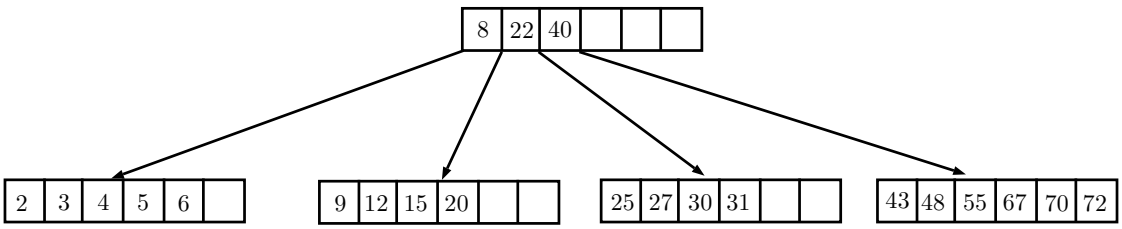

Figura 1: Exemplo de árvore B*

  - imprimir lista de livres do arquivo de índices: imprime a lista de nós livres da árvore devido às remoções
  - imprimir lista de livres do arquivo de dados: imprime a lista de registros livres devido às remoções no arquivo de dados

– realizar operações em lote: realiza operações de inserção, alteração e remoção a partir de arquivo texto a ser carregado, cuja sintaxe é descrito mais adiante.

Toda a interface do programa deve ser apresentada em modo texto, sendo executado no *prompt* do *shell* do sistema operacional.

O conteúdo do arquivo texto para realização de operações em lote, deve ter sintaxe conforme abaixo:
$< operacao >; < codigo >; < campo1 >; < campo2 >; ... < campoN >$
onde:

- $< operacao >$: pode ser `I` (inserção), `A` (alteração) ou `R` (remoção)
- $< campo1 >; < campo2 >; ... < campoN >$ correspondem aos campos com informações referentes ao respectivo item e operação.

Exemplo de conteúdo de arquivo .txt contendo lote de operações a serem executadas:

```
I;70;Relogio smartwatch;Polar;eletronicos e tecnologia;27;566,70
I;25;Leite;Parmalat;bebidas;358;7,70
I;200;Microondas;LG;eletrodomesticos;53;690,99
I;80;Multiprocessador;Arno;eletrodomesticos;7;299,90
I;50;Guarana;Antartica;bebidas;200;5,50
I;30;12 Regras para a Vida: um antídoto para o caos;Alta Books;livro; 54,90
A;25;340;8,30
A;80;5;
A;30;;61,90
I;11;Impressora Laser;HP;eletronicos e tecnologia;15;779,90
I;240;Dom Casmurro;Cia das Letras;livro;30;22,90
I;100;A Condição Humana;Ed. Pensamento; livro;77;96,90
R;50
A;100;72;
I;120;Celular;Apple;eletronicos e tecnologia;25;3200,00
I;90;Suco de laraja;Del Valle; bebidas; 200;9,90
R;200
```

Assim, a linha
`I;70;Relogio smartwatch;Polar;eletronicos e tecnologia;27;566,70`
é interpretada como:

- o código do produto é `70`
- o nome é `Relogio smartwatch`
- a marca é `Polar`
- a categoria é `eletronicos e tecnologia`
- o estoque tem `27` unidades
- o preço unitário é `566,70`.

A linha `A;25;340;8,30` é interpretada como:

- o código do produto é `25`
- o estoque é alterado para `340` unidades
- o preço unitário é alterado para `8,30`

A linha `A;30;;61,90` é interpretada como:

- o código do produto é `30`
- o estoque permanece inalterado

- o preço unitário é alterado para `61,90`

A linha `A;80;5;` é interpretada como:

- o código do produto é `80`
- o estoque é alterado para `5` unidades
- o preço unitário permanece inalterado

Também devem ser observadas as seguintes condições:

- Caso haja tentativa de inserção de um produto com um código já existente, a operação deve ser ignorada.
- Caso haja tentativa de alterar ou remover um produto de código inexistente na árvore, a operação deve ser ignorada.

As informações devem ser armazenadas em 2 arquivos binários conforme mostrado na Figura 2:

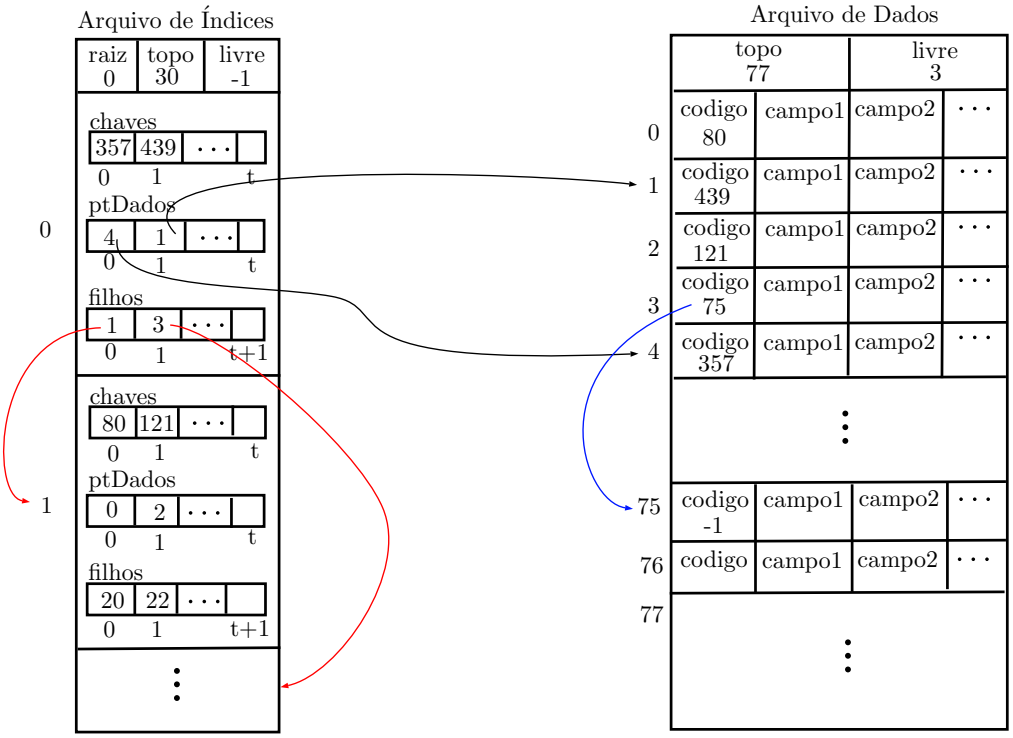

Figura 2: Esquema de arquivos para árvore B*

- um arquivo para os registros de dados: contendo código, nome, marca, categoria, estoque e preço. E um campo adicional para encadeamento de registros livres.
- um arquivo para registros de índices, organizado na forma de árvore B*, sendo o campo código usado como chave. Além de campos para encadeamento dos nós da árvore, devem haver campos para armazenar as posições dos registros de dados no arquivo de dados.

Além disso:

- tanto o arquivo de dados quanto o arquivo de índices devem ser arquivos binários.
- a árvore NÃO DEVE SER CARREGADA INTEIRAMENTE na memória principal, apenas as informações do registro sendo manipulado no momento.
- no início do arquivo de índices (árvore B*) deve haver um cabeçalho contendo o endereço do registro raiz, a posição do topo (primeira posição livre do arquivo) e o endereço da cabeça de nós (páginas) livres.

- no início do arquivo de dados deve haver um cabeçalho contendo o endereço do topo (primeira posição livre do arquivo) e o endereço da cabeça de registros de dados livres
- no caso de remoção de um produto do cadastro, a posição correspondente dentro do arquivo binário deve ser colocada em uma lista de nós livres para reaproveitamento em futuras inserções. Deve existir uma lista de livres para o arquivo de índices e outra para o arquivo de dados. Pode ser reaproveitado algum campo do registro para fazer o encadeamento da lista de livres.
- a implementação deve considerar a possibilidade de escolher a ordem da árvore B* alterando-se em um único local do código-fonte. Para efeitos de teste, iremos considerar ordem 7.
- deve ser implementado o uso da operação de redistribuição de chaves durante a inserção para visando postergar a operação de split.
- a operação de *split* deve seguir o padrão *split 2-to-3*, com exceção da raiz, que deve seguir o *split* tradicional gerando 2 nós.
- o trabalho deverá ser apresentado oralmente em grupo para o professor no horário de aula, constando de execução do programa, seguido de arguição sobre o código/teoria relacionada.

Critérios de avaliação:

- documentação/organização do código-fonte: peso 1
- apresentação oral: peso 1
- implementação correta das funcionalidades: peso 8, assim distribuído:
  - 80% para funcionamento correto
  - 20% para qualidade do código implementado na funcionalidade

  As alterações de preço e quantidade em estoque serão avaliadas como uma única funcionalidade.

Trabalhos copiados ou plagiados receberão nota ZERO!