

Atividade Prática 03

“Manipulação de Árvores Binárias”

Universidade Tecnológica Federal do Paraná (UTFPR), campus Apucarana
Curso de Engenharia de Computação
Disciplina de Estrutura de Dados - ED62A - 1ºSemestre 2019
Prof. Dr. Rafael Gomes Mantovani

1 Descrição

Um “**índice remissivo**” lista os termos e tópicos que são abordados em um documento, juntamente com as páginas em que aparecem. A figura 1 mostra um exemplo de índice remissivo encontrado em livros.

Capone D	264	Dalcin PTR	83, 107, 461, 1079
Cardoso PFG	59, 583, 772, 1040	Dalcin TC	683
Carneiro ACC	900	Danilovic DLS	118
Carpes MF	143	De Capitani EM	294, 817, 822
Caruso P	167, 622	De Martino MC	1033
Carvalho A	245	Defaveri J	595
Carvalho CRR	167, 362, 622	Delfino VDA	907
Carvalho M	745	Detoni VV	607
Castro HA	575	Dias ART	690
Cataneo AJM	595	Dias AS	453
Cataneo DC	595	Dias BA	532, 749
Cau SBA	412	Dias-Junior CA	412
Cavallazzi ACC	34	Dietze R	404, 506, 601, 607
Cavallazzi R	34	D'Império FT	537
Cerci Neto A	103, 639	Dominoni RL	745

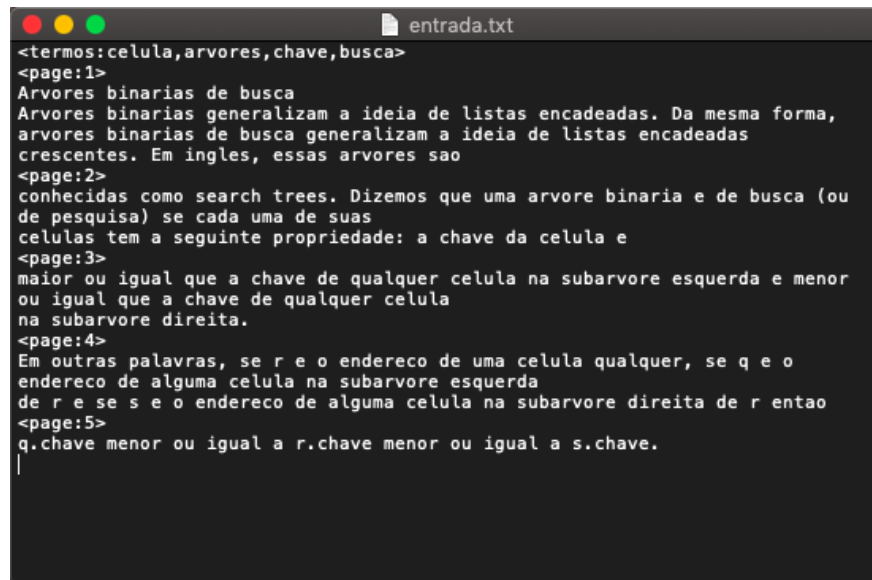
Figura 1: Exemplo de índice remissivo de um livro. Os nomes dos autores são listados, e as correspondentes páginas onde aparecem são apresentadas em sequência.

Desta forma, explore e extrapole as implementações de árvores desenvolvidas em sala para implementar um programa que crie um índice remissivo para um determinado texto de entrada. O programa receberá dois arquivos texto como parâmetros de entrada:

- **arquivo de entrada:** um arquivo texto contendo o texto de entrada simulando um livro e o conteúdo de suas páginas (Figura 2). A primeira linha do arquivo é uma **tag** **<termo:>** contendo todos os termos (palavras) que serão consultados:

$\langle \text{termos: } palavra_1, palavra_2, palavra_3, \dots, palavra_N \rangle$

Na Figura 2, os termos que serão consultados são: **celula**, **arvores**, **chave**, **busca**. Cada palavra terá sua existência verificada no texto todo. Da segunda linha em diante poderão haver diversas tags **<page:X>**, indicando o início de uma página fictícia do livro. Estas tags possuem também o correspondente número de página. Por exemplo: **<page:1>** inicia a apresentação dos textos contidos na página 1; **<page:2>** os textos da página 2, e assim por diante. **Dica:** o texto que aparece entre duas tags de páginas deve ser manipulado como strings, e cada palavra manipulada isoladamente;



```

<termos:celula,arvores,chave,busca>
<page:1>
Arvores binarias de busca
Arvores binarias generalizam a ideia de listas encadeadas. Da mesma forma,
arvores binarias de busca generalizam a ideia de listas encadeadas
crescentes. Em ingles, essas arvores sao
<page:2>
conhecidas como search trees. Dizemos que uma arvore binaria e de busca (ou
de pesquisa) se cada uma de suas
celulas tem a seguinte propriedade: a chave da celula e
<page:3>
maior ou igual que a chave de qualquer celula na subarvore esquerda e menor
ou igual que a chave de qualquer celula
na subarvore direita.
<page:4>
Em outras palavras, se r e o endereco de uma celula qualquer, se q e o
endereco de alguma celula na subarvore esquerda
de r e se s e o endereco de alguma celula na subarvore direita de r entao
<page:5>
q.chave menor ou igual a r.chave menor ou igual a s.chave.
|

```

Figura 2: Exemplo de arquivo de entrada.

- **arquivo de saída:** é o arquivo texto onde serão impressos os correspondentes termos de busca, seguidos dos ids da páginas onde foram encontrados - um termo por linha. Por exemplo: a palavra "chave" aparece nos textos das páginas 2, 3 e 5. Assim, deve-se imprimir no arquivo de saída uma linha contendo: “celula, 2, 3, 5” (ver Figura 3).

Exemplo de arquivos de entrada e saída são apresentados nas Figuras 2 e 3. Na manipulação do programa vocês devem controlar a entrada dos dados usando os argumentos **argc** e **argv** da função **main**. Para executar o programa por linha de comando, deve-se obedecer o seguinte padrão:

<nome do programa> <arquivo de entrada> <arquivo de saída>

Exemplo:

indice entrada.txt saida.txt

2 Orientações gerais

- Implementar também o controle de erros, para lidar com exceções que possam ocorrer (arquivo não aberto, erro de leitura, etc);

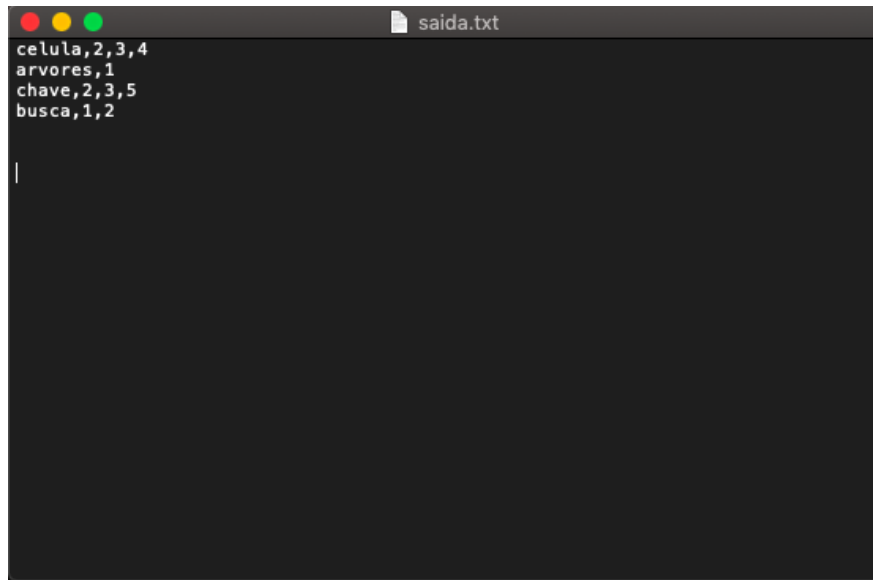


Figura 3: Exemplo de arquivo de saída.

- Para acompanhamento do desenvolvimento, criar um repositório individual com o código desenvolvido no **github Classroom**, por meio do link: <https://classroom.github.com/a/UbyLi4ti>. Os repositórios serão privados, com acesso apenas do professor e do aluno.
- Entrega do programa final: via Moodle. O aluno deve submeter o fonte no link da atividade disponibilizado na página da disciplina no Moodle.
- **Data de entrega: 24/05/2019.**
- Os códigos desenvolvidos por cada aluno serão também verificados por ferramentas de plágio. Códigos iguais/similares terão nota zero.

3 Links úteis

Arquivos em C:

- <https://www.inf.pucrs.br/~pinho/LaproI/Arquivos/Arquivos.htm>
- <https://www.geeksforgeeks.org/basics-file-handling-c/>
- <https://www.programiz.com/c-programming/c-file-input-output>

Argumentos de Linha de comando (argc e argv):

- https://www.tutorialspoint.com/cprogramming/c_command_line_arguments.htm
- <http://linguagemc.com.br/argumentos-em-linha-de-comando/>
- http://www.univasf.edu.br/~marcelo.linder/arquivos_pc/aulas/aula19.pdf

- http://www.inf.ufpr.br/cursos/ci067/Docs/NotasAula/notas-31_Argumentos_linha_comando.html
- <http://www.dca.fee.unicamp.br/cursos/EA876/apostila/HTML/node145.html>

Referências

- [1] Thomas H. Cormen,; Ronald Rivest; Charles E. Leiserson; Clifford Stein. Algoritmos - Teoria e Prática - 3ª Ed. Elsevier - Campus, 2012.
- [2] Nivio Ziviani. Projeto de algoritmos com implementações: em Pascal e C. Pioneira, 1999.
- [3] Adam Drozdek. Estrutura De Dados E Algoritmos Em C++. Cengage, 2010.