

# Atividade Prática 04

## “Manipulação de AVLs”

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

Explore e extrapole as implementações de árvores binárias balanceadas desenvolvidas em sala para implementar um programa que insira e remova chaves inteiras de uma árvore AVL. O programa receberá dois arquivos texto como parâmetros de entrada:

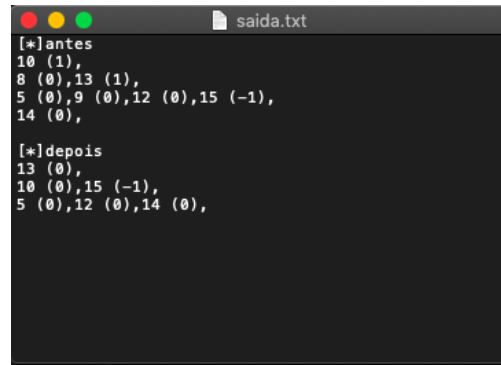
- **arquivo de entrada:** um arquivo texto contendo os valores a serem manipulados pela árvore (Figura 1). A primeira linha do arquivo apresenta em ordem as chaves que precisam ser inseridas na AVL. Em sequência, a segunda linha do arquivo contém as chaves que serão removidas após todas as inserções.



Figura 1: Exemplo de arquivo de entrada.

- **arquivo de saída:** é o arquivo onde serão impressas os estados de configuração da AVL. Neste arquivo vocês devem imprimir duas árvores - a configuração da árvore após todas as inserções; e a configuração da árvore após realizadas as remoções. A Figura 2 mostra as correspondentes árvores geradas pela entrada fornecida na Figura 1.

Na manipulação do programa vocês devem controlar a entrada dos dados usando os argumentos **argc** e **argv** da função **main**. Com isso é possível executar o programa com qualquer entrada que se adeque ao padrão estabelecido. Para executar o programa por linha de comando, deve-se obedecer o seguinte padrão:



```
[*]antes
10 (1),
 8 (0),13 (1),
 5 (0),9 (0),12 (0),15 (-1),
14 (0),

[*]depois
13 (0),
10 (0),15 (-1),
 5 (0),12 (0),14 (0),
```

Figura 2: Exemplo de arquivo de saída.

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

Exemplo:

**avls entrada.txt saida.txt**

onde:

- “avls” é o nome do arquivo da aplicação (avls.c);
- “entrada.txt” é o arquivo de entrada;
- “saida.txt” é o arquivo de saída.

## 2 Orientações gerais

- Implementar também o controle de erros, para lidar com exceções que possam ocorrer (arquivo não aberto, erro de leitura, etc);
- Para acompanhamento do desenvolvimento, criar um repositório individual com o código desenvolvido no **github Classroom**, por meio do link: <https://classroom.github.com/a/eK-4p0eB>. 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: 16/04/2019.**
- 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](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.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.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.