



# Implementação 1

## Árvores Binárias de Busca

### 1 Descrição

Esta implementação consiste em criar um programa que mantenha uma árvore binária de busca, conforme estudado nas aulas da disciplina.

Seu programa deve funcionar como um interpretador de comandos e dar suporte para as seguintes operações:

- **inserir** `<x>` - insere a chave `x` na árvore;
- **buscar** `<x>` - busca pela chave `x` na árvore;
- **remover** `<x>` - remove a chave `x` da árvore;
- **visitar** `<p>` - realiza o percurso `<p>` na árvore. A visita em cada nó apenas imprime o valor da chave;
- **imprimir** - imprime o estado atual da árvore.

### 2 Desenvolvimento

A implementação poderá ser realizada nas linguagens de programação C ou Java, de acordo com a preferência do aluno. De acordo com a linguagem escolhida, o aluno deverá levar em conta a existência dos seguintes arquivos:

- **imp1-c.rar:** destinado aos alunos que realizarão a implementação em C. Esse arquivo compactado contém três arquivos fontes na linguagem de programação C, são eles:
  - **ArvBinBusca.c**  
Arquivo que contém a implementação da função `main`. Nela estará o interpretador de comandos. Esse arquivo não deve ser alterado.
  - **Abb.h**  
Arquivo de cabeçalho que contém a definição do tipo `No` e a prototipação das funções que mantêm uma árvore binária de busca. Esse arquivo também não deve ser alterado.
  - **Abb.c**  
Nesse arquivo serão implementadas as funções que manipulam uma árvore binária de busca. Esse é o arquivo que você deverá implementar. Você notará que nesse arquivo já existem algumas funções totalmente implementadas e outras que são indicadas pelo termo `/*TODO*/`<sup>1</sup>. Essas funções indicadas são

---

<sup>1</sup>do inglês *to do* = fazer

as que deverão ser implementadas. Antes da assinatura dessas funções, há um breve comentário sobre quais parâmetros são recebidos, qual a funcionalidade da função e qual a resposta esperada. Qualquer outra função que você julgue necessária também pode ser criada nesse arquivo.

- **imp1-java.rar**: destinado aos alunos que realizarão a implementação em Java. Esse arquivo compactado contém três arquivos fontes na linguagem de programação Java, são eles:

- **ArvBinBusca.java**

Arquivo que contém a implementação da classe principal do projeto. Nela estará o interpretador de comandos. Essa classe não deve ser alterada.

- **No.java**

Arquivo que contém a implementação da classe que representa um elemento da árvore. Essa classe também não deve ser alterada.

- **Abb.java**

Nesse arquivo será implementada a classe que representa uma árvore binária de busca. Essa é a classe que você deverá implementar. Você notará que nessa classe já existem alguns métodos totalmente implementados e outros que são indicados pelo termo */\*TODO\*/*<sup>2</sup>. Esses métodos indicados são os que deverão ser implementados. Antes da assinatura desses métodos, há um breve comentário sobre quais parâmetros são recebidos, qual a funcionalidade do método e qual a resposta esperada. Qualquer outro método que você julgue necessário também pode ser criado nessa classe.

### 3 Entrega

Instruções para entrega de sua implementação:

#### 1. O que entregar?

Você deve entregar um único arquivo com extensão **.rar** contendo o(s) arquivo(s) com sua implementação. De todos os arquivos que foram disponibilizados, entregue apenas os arquivos que foram modificados por você. Dê o nome do seu usuário do laboratório do CPTL para seu arquivo e adicione a extensão **.rar**. Por exemplo, **ronaldo.santos.rar** é um nome válido.

Lembre-se: seu arquivo compactado deve conter **APENAS** os arquivos fontes **.c** ou **.java** que foram modificados durante a implementação. **NÃO** entregue qualquer outro arquivo, tais como os arquivos fontes que não foram modificados e muito menos os arquivos **.o** ou **.class**, já compilados.

#### 2. Cabeçalho

Todos os arquivos fontes da sua implementação, que forem modificados, devem ter um cabeçalho com o seguinte formato:

---

<sup>2</sup>do inglês *to do* = fazer



```
1  /*****
2  *
3  *  Nome completo
4  *  RGA
5  *  Implementação 1
6  *  Disciplina: Estruturas de Dados e Programação I*
7  *  Professor:  Ronaldo Fiorilo
8  *
9  *****/
```

### 3. Forma de entrega

A entrega será realizada diretamente no *Moodle do CPTL/UFMS*, na disciplina de Estruturas de Dados e Programação I. Para entrega do trabalho, você deve estar cadastrado na página [lives.ufms.br/moodle/](http://lives.ufms.br/moodle/) na disciplina Estruturas de Dados e Programação I. Após abrir uma sessão digitando seu login e sua senha, vá até o tópico Implementações e escolha “Envio - implementação 1”. Você pode submeter sua implementação quantas vezes quiser até às **23 horas** do dia **3 de fevereiro de 2015**. A última versão entregue é aquela que será corrigida. Encerrado o prazo, não serão mais aceitas implementações.

### 4. Atrasos

Implementações atrasadas não serão aceitas. Não deixe para entregar sua implementação na última hora. Para prevenir imprevistos como queda de energia, falha de conexão com a internet ou problemas com o sistema, sugerimos que a entrega da implementação seja feita pelo menos um dia antes do prazo determinado.

### 5. Erros

Os arquivos fontes serão compilados via linha de comando para a correção. Por isso, caso você use algum Ambiente de Desenvolvimento Integrado (IDE), como o Eclipse ou o Netbeans, para desenvolver seu programa, antes de entregá-lo verifique se o seu programa compila **SEM MENSAGENS DE ALERTA** e executa corretamente via linha de comando. Implementações com erros de compilação receberão nota **ZERO**. Faça todos os testes necessários para garantir que seu programa está livre de erros de compilação.

### 6. Verificação dos dados de entrada

Não se preocupe com a verificação dos dados de entrada do seu programa. Considere que somente serão passadas instruções válidas para seu interpretador de comandos e você só precisa se preocupar em tratar esse tipo de instrução.

### 7. Arquivos com o programa fonte

Os arquivos contendo o programa fonte devem estar bem organizados e comentados. Um programa em uma linguagem de programação tem de ser muito bem compreendido por uma pessoa. Verifiquem se o código tem a indentação adequada,



comentários, se não tem linhas muito longas, se tem variáveis com nomes significativos, entre outros detalhes. Não esqueçam que um programa bem descrito e bem organizado é a chave do sucesso.

## 8. Conduta ética

A implementação deve ser feita **INDIVIDUALMENTE**. Cada aluno tem responsabilidade sobre cópias de sua implementação, mesmo que parciais. Não faça a implementação em grupo e não compartilhe seu programa ou trechos de seu programa. Você pode consultar seus colegas para esclarecer dúvidas e discutir idéias sobre o trabalho, ao vivo ou no fórum de discussão da disciplina, mas **NÃO** copie o programa!

Trabalhos considerados plagiados terão nota **ZERO**. O estudante que se envolver em **DOIS CASOS DE PLÁGIO** estará automaticamente **REPROVADO** na disciplina.