UNIVERSIDADE FEDERAL DO PIAUÍ - UFPI
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS
Curso Bacharelado em Sistemas de Informação
Disciplina: Estrutura de Dados II
Docente: Juliana Oliveira de Carvalho
Discente: Erlanny Rodrigues da Silva Rêgo

Relatório Trabalho II

Resumo

Este trabalho apresenta a implementação de um sistema em C para gerenciamento de dados geográficos e populacionais do Brasil, utilizando estruturas de dados avançadas. O sistema combina listas duplamente encadeadas para armazenar estados em ordem alfabética, árvores vermelhas-pretas para organizar cidades e CEPs, e árvores 2-3 para cadastrar pessoas, garantindo eficiência nas operações de inserção, busca e remoção. As árvores vermelhas-pretas mantêm o balanceamento automático através de rotações e trocas de cores, enquanto as árvores 2-3 asseguram a ordenação e o equilíbrio mesmo com alterações frequentes. O sistema inclui validações de integridade, como a vinculação correta entre cidades e estados e a restrição de remoção de CEPs apenas quando não associados a pessoas. Os resultados demonstram que as estruturas escolhidas oferecem desempenho eficiente (O(log n)) e organização adequada dos dados. A implementação serviu como exemplo prático da aplicação dessas estruturas em problemas reais, cumprindo os objetivos propostos com robustez e clareza na manipulação das informações.

1 Introdução

As estruturas de dados desempenham um papel importante na computação, permitindo o armazenamento e gerenciamento eficiente de informações. Entre as diversas abordagens, as árvores balanceadas se destacam por manter operações rápidas mesmo com grandes volumes de dados. Este trabalho explora duas estruturas fundamentais: a Árvore Vermelha-Preta e a Árvore 2-3, aplicando-as na construção de um sistema para armazenar e gerenciar dados geográficos.

O projeto desenvolvido em linguagem C combina diferentes estruturas para criar uma solução robusta e eficiente. Utiliza listas duplamente encadeadas para organizar os estados brasileiros em ordem alfabética, enquanto emprega Árvores Vermelha-Pretas e Árvores 2-3 para gerenciar as cidades e seus respectivos CEPs e armazenar os registros das pessoas, garantindo o balanceamento automático através de operações. O sistema foi projetado com atenção às regras de integridade, assegurando que cada cidade esteja vinculada a um estado existente e que os CEPs só possam ser removidos quando não associados a nenhuma pessoa.

Os resultados obtidos demonstram a eficácia das estruturas escolhidas, mostrando como as Árvores Vermelha-Pretas e 2-3 mantêm os dados balanceados e acessíveis, mesmo após diversas operações de inserção e remoção. A implementação não apenas cumpre os requisitos propostos, mas também serve como exemplo prático de como estruturas de dados

avançadas podem resolver problemas complexos de organização e recuperação de informações.

2 Seções Específicas

Nesta seção, serão apresentadas as partes específicas do trabalho. Primeiramente, é fundamental compreender um dos principais mecanismos da árvore Vermelha-Preta e 2-3: as rotações, que são utilizadas para manter o balanceamento da estrutura. Abaixo, será feita uma breve explicação sobre esse conceito, seguida pela descrição dos problemas propostos e suas respectivas soluções, utilizando as estruturas de dados mencionadas anteriormente.

2.1 Árvore Vermelha-Preta

Uma Árvore Vermelha-Preta deve satisfazer as seguintes propriedades:

- **Cor**: Cada nó é vermelho ou preto. Essa cor é armazenada em um campo adicional no nó.
- **Raiz**: O nó raiz da árvore é sempre preto, independentemente de qualquer operação.
- **NULLs**: Todos nós nulos são pretos.
- **Vermelha**: Se um nó é vermelho, então ambos seus filhos são pretos. Isso significa que não podem existir dois nós vermelhos consecutivos em qualquer caminho da raiz até uma folha.
- **Caminho**: Para cada nó, todos os caminhos simples deste nó até suas folhas descendentes contêm o mesmo número de nós pretos.
- **Novo nó**: Todo nó recém-criado (inserido) é inicialmente vermelho. Após a inserção, podem ser necessários ajustes para manter as propriedades.

2.1.1 Rotações na Árvore Vermelha-Preta

As rotações são operações fundamentais para manter o balanceamento da árvore após inserções e remoções.

Rotação Simples

A rotação simples pode ser para a esquerda ou para a direita. Quando fazemos uma rotação à direita, pegamos um nó que está desbalanceado para a esquerda e o "empurramos" para baixo à direita, fazendo com que seu filho esquerdo suba para ocupar sua posição original. A rotação à esquerda é o processo inverso (Figura 1).

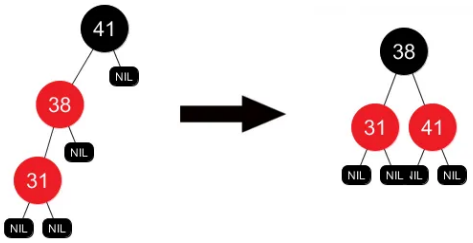

Figura 1 - Rotação simples da vermelha-preta.

Rotação Dupla

As rotações duplas são necessárias quando uma única rotação não resolve o desbalanceamento. Uma rotação esquerda-direita consiste primeiro em uma rotação à esquerda no filho esquerdo do nó desbalanceado, seguida por uma rotação à direita no próprio nó desbalanceado. Já a rotação direita-esquerda começa com uma rotação à direita no filho direito, seguida por uma rotação à esquerda no nó problemático. Como mostra a Figura 2.

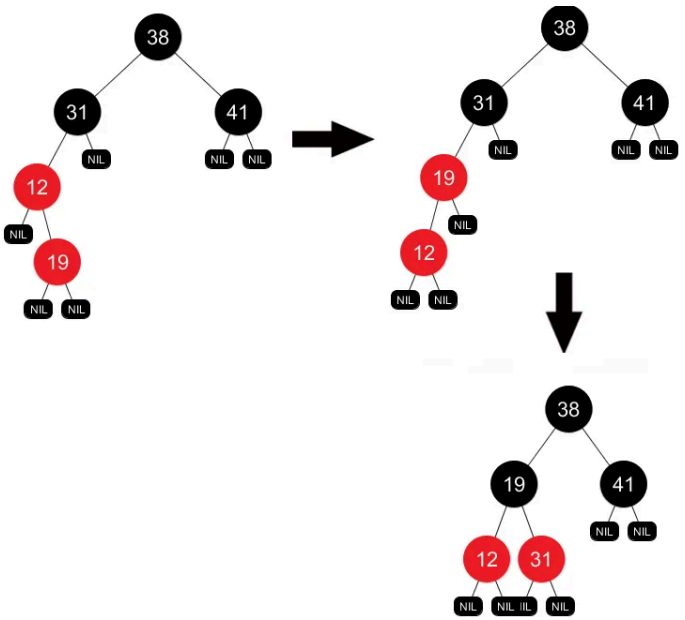

Figura 2 - Rotação dupla da vermelha-preta.

Troca cor

A troca de cores simplesmente transforma nós vermelhos em pretos e vice-versa, tanto no nó em questão quanto em seus filhos imediatos, reequilibrando as propriedades da árvore (Figura 3).

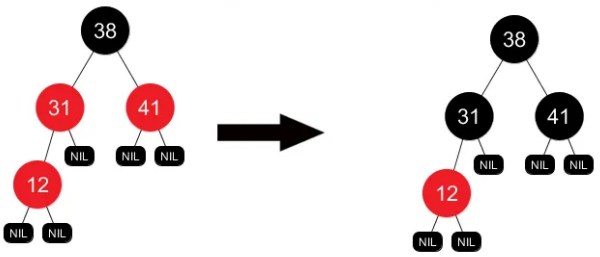

Figura 3 - Troca cor da vermelha preta.

2.2 Árvore 2-3

Uma árvore 2-3 é uma estrutura de dados do tipo árvore de busca balanceada onde cada nó pode ter: 2 filhos e 1 chave (nó 2) ou 3 filhos e 2 chaves (nó 3). Essa estrutura garante que a árvore sempre permaneça balanceada, mantendo operações de inserção, remoção e busca eficientes com complexidade O(log n).

Em um nó 2, que contém uma única chave K, a subárvore à esquerda armazena chaves menores que K, enquanto a subárvore à direita contém chaves maiores que K. Já em um nó 3, que possui duas chaves K1 e K2 (com K1 < K2), a subárvore à esquerda guarda chaves menores que K1, a subárvore do meio armazena chaves entre K1 e K2, e a subárvore à direita contém chaves maiores que K2 (Figura 4).

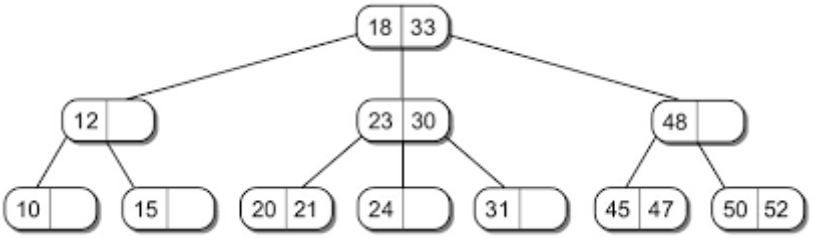

Figura 4 - Árvore 2-3.

Durante a inserção, a operação sempre ocorre em um nó folha. Se um nó atingir o limite de três chaves, ele é dividido, promovendo a chave do meio para o nó pai, garantindo assim a manutenção das propriedades da árvore. Na remoção, pode ser necessária a redistribuição de chaves ou a fusão de nós para preservar o balanceamento, assegurando que todos os nós folha permaneçam no mesmo nível.

2.3 Estrutura do trabalho

A estrutura do trabalho está organizada em uma pasta principal chamada “trabalho-II”, que contém duas subpastas: uma referente à árvore Vermelha-Preta e outra à árvore 2-3. Cada uma dessas subpastas possui uma pasta chamada “includes”, onde estão localizados os arquivos de cabeçalho com as declarações das funções e

estruturas de dados utilizadas, e uma pasta “src”, que contém as implementações dessas funções. Como mostra a Figura 5.

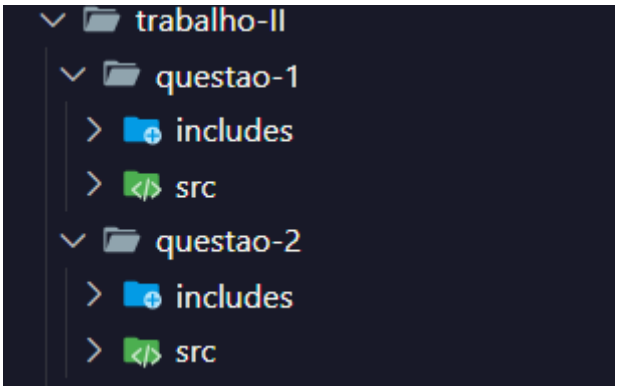

Figura 5 - Estrutura do trabalho

Para executar o projeto, basta compilar e rodar o arquivo executável “main.exe” diretamente no terminal, dentro do diretório correspondente à árvore desejada, como por exemplo em “../questao-1/src/” para a Vermelha-Preta e “../questao-2/src/” para a 2-3.

2.4 Problema

Desenvolva um programa em C que utilize estruturas de dados interligadas para representar informações geográficas e populacionais do Brasil. O sistema deve conter:

- Uma lista duplamente encadeada de Estados, ordenada por nome, contendo dados como nome do estado, capital, número de cidades, população total e um ponteiro para uma estrutura de cidades.
- Cada cidade possui nome, população e um conjunto de CEPs.
- Cada CEP é armazenado individualmente e associado à cidade correspondente.
- Uma estrutura separada deve conter dados de pessoas, incluindo CPF, nome, CEP de nascimento, CEP de residência e data de nascimento.

O sistema deve permitir cadastrar, consultar e remover dados em qualquer momento, com restrições lógicas de integridade (ex: só cadastrar cidade se o estado existir, só remover CEP se não estiver associado a ninguém).

2.4.1 Lista Duplamente Encadeada

Neste tópico, descrevemos a funcionalidade de cada função relacionada à lista duplamente encadeada, implementada no arquivo estado.c.

- criarNoEstado: Cria e inicializa um novo nó do tipo Estado com o nome do estado e da capital, e valores padrões para os demais campos. Retorna um ponteiro para o novo estado.

- insertionSort: Ordena a lista duplamente encadeada de estados em ordem alfabética pelo nome do estado, usando o algoritmo Insertion Sort adaptado para listas.
- inserirEstadosOrdenado: Insere um novo estado no final da lista e, em seguida, ordena toda a lista em ordem alfabética usando a função insertionSort.
- buscaEstado: Percorre a lista de estados e, se encontrar um estado com o nome fornecido, armazena seu ponteiro em resultado.
- imprimirEstados: Imprime na tela o nome, capital, população e número de cidades de todos os estados da lista.

2.4.2 Árvore Vermelha Preta

Neste tópico, descrevemos a funcionalidade de cada função relacionada à árvore vermelha-preta, implementada no arquivo cidade.c.

- corCidade: Retorna a cor de um nó da árvore de cidades. Se o ponteiro for nulo, retorna a cor BLACK.
- rotacaoEsqCidade: Realiza uma rotação à esquerda na árvore rubro-negra, ajustando ponteiros e cores conforme necessário.
- rotacaoDirCidade: Realiza uma rotação à direita na árvore rubro-negra, ajustando ponteiros e cores conforme necessário.
- trocarCorCidade: Inverte a cor do nó atual e de seus filhos esquerdo e direito, caso existam, usado no balanceamento da árvore rubro-negra.
- buscaCidade: Busca uma cidade na árvore binária de cidades com base no nome, armazenando o ponteiro da cidade encontrada em resultado.
- criarNoCidade: Cria um novo nó de cidade com nome, população, cor RED e ponteiros nulos.
- insereCidade: Insere um nó de cidade na árvore rubro-negra, mantendo a ordenação por nome e ajustando o balanceamento com rotações e troca de cores.
- cadastrarCidade: Busca um estado pelo nome e insere uma nova cidade nele. Atualiza o total de cidades e população. Se for a capital, define o ponteiro capital.
- insereAjustaCor: Insere a cidade e, se a inserção for bem-sucedida, garante que a raiz da árvore fique com a cor BLACK. Exibe uma mensagem de sucesso.
- imprimirCidades: Imprime recursivamente os nomes das cidades em ordem alfabética, com sua população e cor (usando travessia in-order).
- imprimirEstadosCidadesCeps: Imprime os dados dos estados (nome, capital, população, número de cidades) e chama a função para imprimir suas cidades e respectivos CEPs.

2.4.3 Árvore 2-3

Neste tópico, descrevemos a funcionalidade de cada função relacionada à árvore vermelha-preta, implementada no arquivo cidade.c.

- criar_pessoa: Cria uma nova estrutura Pessoa a partir de informações passadas (nome, CPF, data de nascimento, CEPs) e retorna a pessoa criada.
- criarNo: Cria um novo nó da árvore 2-3, contendo uma pessoa, e o inicializa como uma folha sem filhos.
- inserir: Insere uma nova pessoa na árvore 2-3, mantendo a estrutura balanceada com redistribuição e divisões de nós quando necessário.
- buscar: Procura por uma pessoa na árvore 2-3 com base no CPF e retorna a estrutura Pessoa correspondente, se encontrada.
- imprimir: Imprime os dados de todas as pessoas armazenadas na árvore, em ordem, percorrendo recursivamente os nós.
- remover: Remove uma pessoa da árvore com base no CPF, ajustando a estrutura da árvore após a remoção para manter o balanceamento.
- liberar_arvore: Libera toda a memória alocada para a árvore 2-3, incluindo os nós e os dados das pessoas.

2 Resultados da Execução do Programa

Esta seção apresenta exemplos práticos da execução do programa, demonstrando suas funcionalidades. A Figura 6 mostra o menu principal, usado tanto para a árvore vermelha-preta quanto para a árvore 2-3. O menu inclui opções para cadastrar estados, cidades, CEPs e pessoas, além de exibir a lista de estados, cidades e CEPs de forma integrada. Também permite visualizar as pessoas cadastradas, remover CEPs e excluir registros de pessoas. Por fim, a última opção direciona para um menu de curiosidades, ilustrado na Figura 7, que apresenta informações adicionais sobre os dados armazenados.

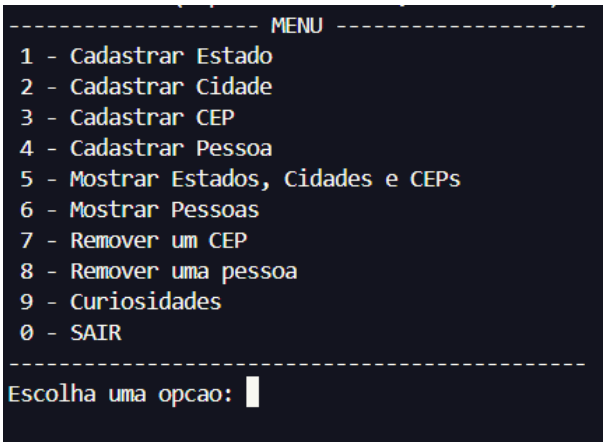

Figura 6 - Menu principal.

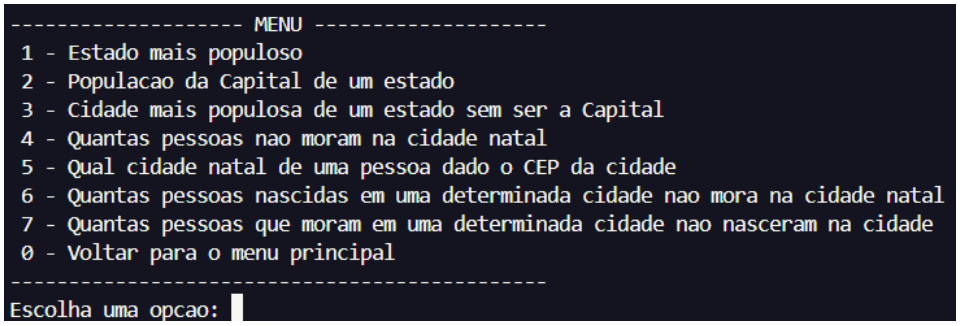

Figura 7 - Menu das curiosidades.

Após o cadastro completo de estados, cidades, CEPs e pessoas, o programa oferece as opções 5 e 6 no menu principal para visualizar todas as informações armazenadas. A opção 5 exibe os dados contidos nas árvores vermelha-pretas, que representam os estados, as cidades e os CEPs cadastrados, permitindo observar a estrutura hierárquica e o balanceamento automático proporcionado por esse tipo de árvore. A Figura 8 apresenta esses dados, demonstrando a organização e o conteúdo das árvores implementadas. Já a opção 6 imprime a árvore vermelha-preta de pessoas, exibindo todos os registros cadastrados em ordem. A Figura 9 mostra essa impressão, evidenciando como a estrutura 2-3 mantém os dados organizados de forma eficiente e balanceada para inserções e buscas.

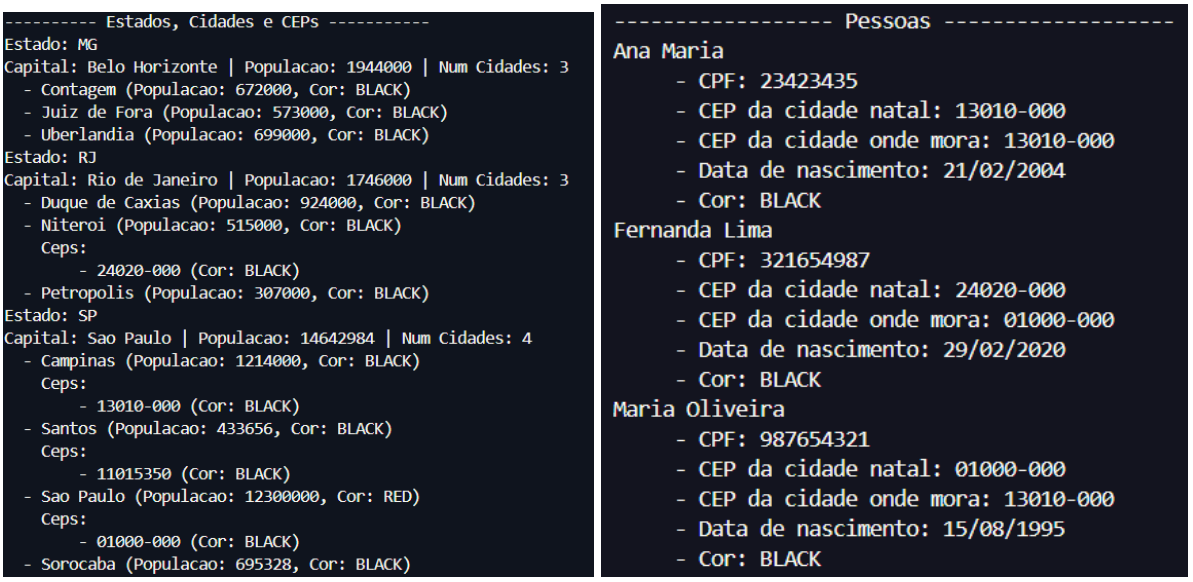

Figura 8 - Impressões da vermelha-preta.

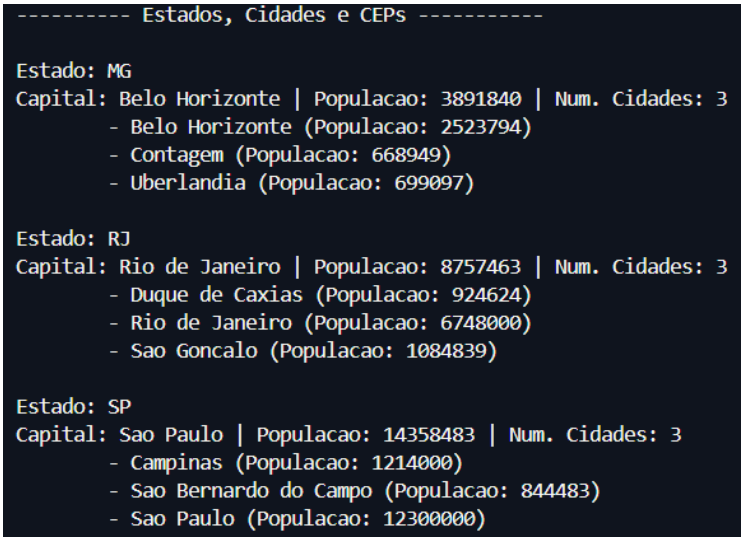

Figura 9 - Impressões da 2-3.

Esses resultados evidenciam a funcionalidade e eficiência das estruturas de dados implementadas no sistema. A utilização de árvores vermelho-pretas para armazenar e organizar estados, cidades e CEPs garante um acesso rápido e estruturado às informações geográficas. Da mesma forma, a árvore 2-3 utilizada para armazenar os dados das pessoas demonstra um desempenho satisfatório tanto na inserção quanto na busca e exclusão de registros, mantendo a ordenação e o balanceamento da estrutura. Dessa forma, o programa se mostra eficaz no gerenciamento de dados complexos, oferecendo uma interface clara para o usuário e garantindo integridade e desempenho na manipulação das informações.

3 Conclusão

O desenvolvimento do sistema proposto demonstrou, na prática, a importância e a eficiência das estruturas de dados avançadas, como as Árvores Vermelha-Preta e as Árvores 2-3, na organização e gerenciamento de grandes volumes de informações. Ao integrá-las com listas duplamente encadeadas, foi possível criar um sistema robusto, capaz de representar a estrutura geográfica e populacional do Brasil de forma eficiente e coesa.

A escolha por utilizar árvores balanceadas permitiu garantir desempenho consistente nas operações de inserção, busca e remoção, mesmo à medida que os dados cresciam. A Árvore Vermelha-Preta mostrou-se adequada para organizar hierarquicamente estados, cidades e CEPs, assegurando o acesso rápido e o balanceamento automático das informações. Por sua vez, a Árvore 2-3 demonstrou grande eficiência no armazenamento dos dados das pessoas, preservando a integridade estrutural mesmo após diversas alterações.

Em resumo, o trabalho cumpriu seus objetivos, servindo como um exemplo funcional do poder das estruturas de dados na resolução de problemas reais. A implementação em linguagem C reforça o domínio de conceitos fundamentais de programação e estruturas de dados, além de evidenciar a aplicabilidade dessas técnicas em contextos complexos e multidimensionais.

Referências

LUKE, S. Red Black Tree — Estrutura de Dados. Medium, 2020. Disponível em: https://medium.com/@luksrn/red-black-tree-estrutura-de-dados-4e13dda71280. Acesso em: [dia] [mês abreviado]. [ano].

MELO, T. Árvores 2-3. tiagodemelo.info, 2020. Disponível em: https://tiagodemelo.info/wp-content/uploads/2020/10/arvores-2-3.pdf.