

ALUNOS: Felipe Gabriel Soares Rodrigues, Cauã Martins da Silva.

PROFESSOR: Raimundo Moura.

CURSO: Ciência da computação - Bacharelado.

COMPONENTE CURRICULAR: Estruturas de dados.

RELATÓRIO TÉCNICO REFERENTE A BST COM INTERFACE GRÁFICA

OBJETIVO: Este relatório tem como finalidade detalhar e explicar o funcionamento de uma árvore binária de busca com interface gráfica, onde o usuário pode acrescentar nomes, números e letras — embora a atividade especifique apenas ‘nomes’ — e acompanhar a formação da árvore pela interface, podendo também ter acesso ao maior/menor elemento, checar a altura da árvore, profundidade de determinado nó, largura interna e até mesmo conferir se está balanceada ou não.

ACERCA DO CÓDIGO: O programa foi implementado em Python, utilizando as bibliotecas Tkinter e collections para, respectivamente, fazer uso de uma fila de duas pontas (necessária para a pesquisa em largura) e criar interfaces gráficas de usuário (GUIs).

A árvore binária utilizada é do tipo SIMPLES, alocando os elementos digitados para seus lugares — à esquerda se menor ou à direita se maior que o nó pai — de acordo com a ordem em que o usuário digitá-los. Além disso, o balanceamento depende INTEIRAMENTE da ordem e dos elementos digitados, uma vez que o código por si só não promove balanceamento, apenas informa a situação do mesmo (se está ou não balanceada).

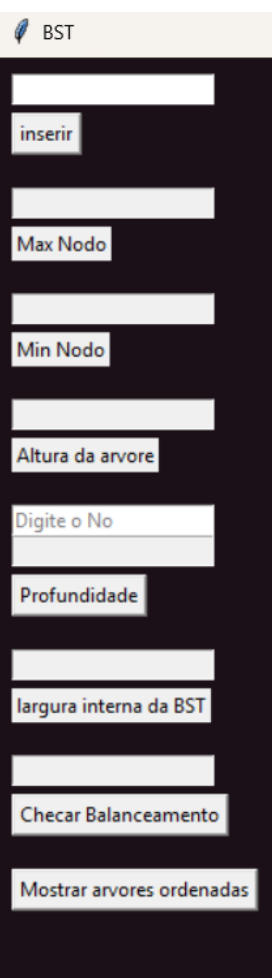
EXPLICAÇÃO DA INTERFACE:

INSERIR: Local onde o usuário deve digitar os números a serem adicionados na árvore. É altamente recomendado não misturar diferentes tipos de entrada, formando árvores binárias de cada tipo por vez (sejam palavras, caracteres ou números), uma vez que embora ordene normalmente, o resultado obtido pode não satisfazer o que o usuário espera receber.

MAX NODO: Indica qual o maior elemento introduzido na BST até o momento, em ordem numérica crescente ou alfabética, dependendo do tipo de dado utilizado. O resultado atual é comparado com cada novo elemento inserido e substituído por ele caso este seja maior.

MIN NODO: Faz exatamente a mesma coisa que o campo anterior, embora em ordem decrescente, ou seja: Ele exibe o menor elemento presente na lista e é substituído caso um novo menor elemento seja introduzido.

ALTURA DA ÁRVORE: Mostra para o usuário a altura da árvore binária, calculando a quantidade de links entre nós não nulos do maior galho existente. Essa altura é atualizada instantaneamente caso haja alterações na altura anterior após uma nova inserção.



The image shows a screenshot of a graphical user interface (GUI) for a Binary Search Tree (BST) application. The interface is titled 'BST' in the top left corner. It features several input fields and buttons arranged vertically. The buttons are labeled: 'inserir', 'Max Nodo', 'Min Nodo', 'Altura da arvore', 'Profundidade', 'largura interna da BST', 'Checar Balanceamento', and 'Mostrar arvores ordenadas'. Each button is associated with an input field above it, except for 'Mostrar arvores ordenadas' which is at the bottom without a direct input field above it.

PROFUNDIDADE: Este campo pede para que o usuário indique o elemento presente no nó em que se deseja medir a profundidade, para então somar a quantidade de links deste nó até a raiz. Caso nada seja digitado, “None” será impresso no campo.

LARGURA INTERNA DA BST: Calcula a maior quantidade de nós presentes em qualquer nível específico da árvore, ou seja: mede a quantidade máxima de nós que aparecem em um mesmo nível.

CHECAR BALANCEAMENTO: Vê se a árvore está balanceada, imprimindo “True” caso sim e “False” caso não. É importante frisar novamente que apesar de descobrir se está ou não balanceada, NÃO será feito nenhum balanceamento caso não esteja.

MOSTRAR ÁRVORES ORDENADAS: Exibe quatro diferentes ordens para os elementos inseridos na BST, pré-ordem, pós-ordem, em-ordem e em-nível. Os quatro tipos de travessia são impressos juntos e ao mesmo tempo, sem que o usuário precise escolher um deles.

INFORMAÇÕES ADICIONAIS:

O que acontece com elementos repetidos?

Não serão aceitos. Caso o usuário digite um elemento já existente na BST, ele será ignorado.

Por que não misturar os elementos?

É impossível encontrar uma equivalência entre números e letras, uma vez que letras são limitadas e números infinitos. Nós sabemos que A é a primeira letra do alfabeto, mas 1 não é o primeiro número, uma vez que não há “primeiro” ou “último”, simplesmente “maior” ou “menor”. O programa aceitará elementos mistos e seguirá normalmente a ordem alfabética ou numérica dependendo do elemento, mas pode ser que a lógica exposta não seja entendida pelo usuário. (qualquer número digitado, seja 1, 1.000 ou 1.000.000, SEMPRE será considerado menor que qualquer letra).

Como criar uma nova árvore?

É preciso fechar a página e executar o programa novamente.

Consigo deletar elementos?

Não. Caso haja algum erro de digitação, será preciso recomeçar.