Identificação: Pedro Mesquita Maia – 2312664

Descrição: O objetivo do trabalho foi implementar uma árvore binária de busca, onde a ideia principal era inserir e ordenar nós com base numa lista de identificadores inteiros. Depois de construir a árvore, a tarefa era fazer uma travessia in-order, pra mostrar os valores em ordem crescente, o que basicamente verifica se a árvore foi montada corretamente.

Estrutura: Criei três funções principais, além da função main, pra dar conta do trabalho: createNode, createBT e showBT. Primeiramente, defini um array de identificadores inteiros. Aí, chamei a função createBT pra construir a árvore binária com esses valores. No final, usei a showBT pra fazer a travessia da árvore e imprimir os identificadores na ordem correta, garantindo que a árvore estava, de fato, organizada.

Solução: Usei um array fixo com os valores {10, 5, 15, 3, 7, 13, 20, 1, 4, 6} pra preencher a árvore. As funções criadas foram:

createNode: Aloca dinamicamente um novo nó, inicializa os ponteiros dos filhos como NULL e define o valor do nó com o identificador passado.
createBT: Essa é a função que constrói a árvore binária de busca. Ela recebe como entrada um ponteiro pra árvore e o array de identificadores. O primeiro valor vira a raiz e depois, com a ajuda de uma fila, vai inserindo os filhos esquerdo e direito dos nós, um por um, até que todos os elementos estejam inseridos. Descrição passo-a-passo:

Inicialização:

A função começa verificando se o array de identificadores está vazio. Se estiver, retorna.
Se não estiver vazio, o primeiro valor do array (10) é usado para criar a raiz da árvore. Esse nó é atribuído a *btPointer e adicionado à fila. Inicialmente, a fila está vazia, então front e rear são ambos 0.

Inserção na Árvore:
O nó da raiz (valor 10) é adicionado à fila. front e rear são atualizados para refletir isso. Neste ponto:
queue[0] contém o nó com valor 10.
front é 0 e rear é 1.

Processamento da Fila:
A partir daqui, a função entra em um loop onde processa cada nó da fila para inserir seus filhos:

Primeira Iteração:
O nó atual é o primeiro nó da fila, com valor 10 (queue[front]), então:
current é o nó com valor 10.
front é 0, e rear é 1.
O próximo valor no array é 5, então 5 é inserido como o filho esquerdo do nó com valor 10. Este nó é então adicionado à fila.
Atualiza a fila:
queue[0] ainda é o nó com valor 10.
queue[1] agora é o nó com valor 5.
rear é 2.
front é 1 após o processamento do nó atual.
Segunda Iteração:

O nó atual é agora o nó com valor 5 (queue[front]):
current é o nó com valor 5.
front é 1, e rear é 2.
O próximo valor no array é 15, então 15 é inserido como o filho direito do nó com valor 10. Este nó é adicionado à fila.
Atualiza a fila:
queue[1] é o nó com valor 5.
queue[2] é o nó com valor 15.
rear é 3.
front é 2 após o processamento do nó atual.
Terceira Iteração:

O nó atual é agora o nó com valor 15 (queue[front]):
current é o nó com valor 15.
front é 2, e rear é 3.
O próximo valor no array é 3, então 3 é inserido como o filho esquerdo do nó com valor 5. Este nó é adicionado à fila.
Atualiza a fila:
queue[2] é o nó com valor 15.
queue[3] é o nó com valor 3.
rear é 4.
front é 3 após o processamento do nó atual.
Quarta Iteração:

O nó atual é agora o nó com valor 3 (queue[front]):
current é o nó com valor 3.
front é 3, e rear é 4.
O próximo valor no array é 7, então 7 é inserido como o filho direito do nó com valor 5. Este nó é adicionado à fila.
Atualiza a fila:
queue[3] é o nó com valor 3.
queue[4] é o nó com valor 7.
rear é 5.
front é 4 após o processamento do nó atual.
Continuação:

O processo continua da mesma maneira, onde o próximo valor no array é inserido como o filho esquerdo ou direito do nó atual, conforme disponível, e o nó inserido é adicionado à fila. Isso ocorre até que todos os valores do array tenham sido processados e a árvore esteja completamente construída.
.

showBT: Printa a árvore. Essa função chama a si mesma de forma recursiva: primeiro vai pro filho da esquerda, depois imprime o nó atual e, por último, vai pro filho da direita.
Na função main, a árvore foi inicializada e depois as funções foram chamadas na sequência pra construir e mostrar a árvore.

Observações: Foi bem desafiador manter a ordem correta na hora de inserir os nós, e a fila ajudou bastante nisso, já que permitiu ir inserindo os filhos na ordem certa. Mas para ser sincero eu não entendi 100%, pra mim ainda não é fluído e eu precisei consultar fórums, vídeos e pedir ajuda para o chatGPT. Acho que como o tempo foi curto (tive aula até ás 19h) eu não consegui fazer tudo sozinho.

Compilação e Testes: Pra compilar o código e verificar possíveis warnings, usei o comando:
gcc -Wall -o main main.c. Fiz testes com a árvore usando essa lista de identificadores e também experimentei outras listas menores, só pra garantir que estava funcionando. O comportamento foi o esperado em todos os testes, sem erros ou problemas de execução.