Aluno: Thiago Pereira Camerato

Matrícula: 2212580

Introdução: O programa desenvolvido tem como objetivo selecionar dois competidores, A e C, que irão competir com o competidor B em rodadas de natação. A seleção é baseada na altura dos competidores, seguindo critérios específicos. O competidor A deve ser o mais alto entre os que são mais baixos que B, e o competidor C deve ser o mais baixo entre os que são mais altos que B. Para isso, foi utilizada uma Árvore Binária de Busca (ABB) para armazenar as alturas dos competidores e realizar as seleções de forma eficiente.

Estrutura do Programa:

O programa consiste em três arquivos **trabalho2_main.c**, **trabalho2_aux.c**, e **trabalho2_aux.h.**

- **trabalho2_main.c** é o arquivo principal que contém a função **main** onde o programa começa sua execução. Ele é responsável por criar a ABB, inserir os competidores na árvore, solicitar a altura do competidor B ao usuário e chamar a função **findCompetitors** para realizar a seleção dos competidores A e C. Também imprime as alturas dos competidores em ordem e a quantidade de nós visitados.
- **trabalho2_aux.c** contém as implementações das funções auxiliares necessárias para a manipulação da ABB e a seleção dos competidores. Estas funções incluem a inicialização da árvore, inserção de nós, busca dos competidores A e C, impressão dos nós em ordem e liberação de memória.
- **trabalho2_aux.h** contém as declarações das estruturas de dados e protótipos de funções utilizadas no programa.

Solução:

A seguir, descreveremos passo a passo as funções implementadas no arquivo **trabalho2_aux.c**, explicando seu funcionamento:

abb* iniciatetree(void):

- Esta função inicia uma Árvore Binária de Busca (ABB) vazia.
- Retorna **NULL** para indicar uma árvore vazia.

abb* createNode(int value):

- Cria um novo nó da ABB com um valor inteiro especificado.
- Aloca memória para o novo nó, atribui o valor e inicializa os ponteiros esquerdo e direito como nulos.
- Retorna o ponteiro para o novo nó criado.

abb* insertNode(abb* tree, int value):

- Insere um novo nó com um valor especificado na ABB.
- Recursivamente, percorre a árvore até encontrar a posição correta para inserção com base nas comparações de valores.

- Retorna a árvore com o novo nó inserido.

abb* max_min_value(abb* tree, int mode):

- Encontra e retorna o nó com o valor máximo (mode = 1) ou mínimo (mode = 0) na ABB.
- Percorre a árvore à direita (mode = 1) ou à esquerda (mode = 0) até encontrar o nó desejado.

abb* predecessor_successor(abb* tree, int mode, int value):

- Encontra e retorna o predecessor (mode = 0) ou sucessor (mode = 1) de um nó com um valor especificado na ABB.
- Percorre a árvore enquanto compara os valores até encontrar o nó desejado.

void findCompetitors(abb* tree, int infoB, int* visitedNodes):

- Seleciona os competidores A e C com base na altura do competidor B.
- Utiliza a ABB para encontrar os competidores de acordo com os critérios definidos.
- Imprime os competidores selecionados e contabiliza o número de nós visitados.

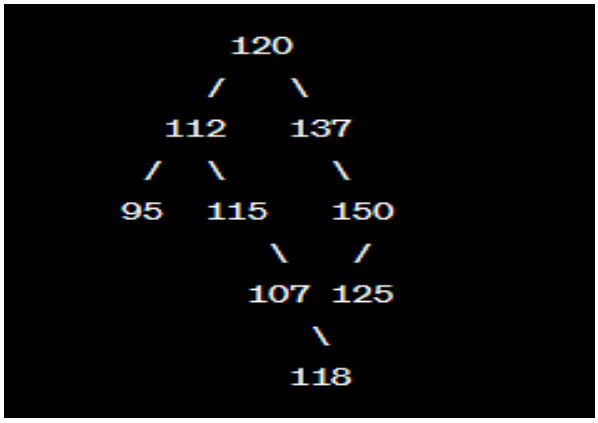
void printTree(abb* tree):

- Imprime as alturas dos competidores armazenados na ABB em ordem crescente.
- **void freeTree(abb* tree)**:
- Libera a memória alocada para todos os nós da ABB, incluindo os nós filhos, de forma recursiva.

Observações e Conclusões:

- A implementação da ABB e das funções auxiliares permite uma busca eficiente dos competidores A e C com um número mínimo de nós visitados.
- A estrutura da ABB é adequada para armazenar e buscar as alturas dos competidores de forma eficiente, o que é essencial para a seleção dos competidores.
- As mensagens de "Competição inválida" são exibidas corretamente quando não é possível encontrar competidores que atendam aos critérios.
- Em geral, a solução implementada atende aos requisitos do problema de forma eficiente e clara, proporcionando uma seleção justa dos competidores de acordo com os critérios estabelecidos.

Árvore após as inserções:



Output do programa:

- **Teste 1:**

```
$ ./trabalho2.exe
Alturas dos competidores em ordem:
95 107 112 115 118 120 125 137 150

Digite a altura do competidor B:
112
Competitor B: 112
Competitor A: 107
Competitor C: 115

Numero de nos visitados: 2
```

- **Teste 2:**

```
$ ./trabalho2.exe
Alturas dos competidores em ordem:
95 107 112 115 118 120 125 137 150

Digite a altura do competidor B:
150
Competitor B: 150
Competitor A: 137
Competition invalid

Numero de nos visitados: 3
```

- **Teste 3:**

```
$ ./trabalho2.exe
Alturas dos competidores em ordem:
95 107 112 115 118 120 125 137 150

Digite a altura do competidor B:
107
Competitor B: 107
Competitor A: 95
Competitor C: 112

Numero de nos visitados: 4
```