Questionário Árvores Rubro-negras

Total de pontos 100/100

O e-mail do participante (**martinson.freitas@gmail.com**) foi registrado durante o envio deste formulário.

✓ 1. Qual das seguintes é uma das principais razões para a popularidade das árvores rubro-negras? *10/10

- ( ) Elas dependem de randomização para suas garantias de tempo de execução.
- ( ) Elas são as mais fáceis de implementar de todas as árvores balanceadas.
- ( ) Suas operações de adição e remoção são sempre O(1) no pior caso.
- ( ) Elas são utilizadas apenas em nichos de aplicação específicos, não em bibliotecas amplamente usadas
- (•) Elas fornecem altura de no máximo 2logn e operações de add(x) e remove(x) em tempo O(logn) no pior caso. ✓

✓ 2. Qual é a altura máxima garantida para uma árvore rubro-negra com n elementos? *10/10

- ( ) n.
- (•) 2log n. ✓
- ( ) log2 n + 2.
- ( ) log n.
- ( ) log3/2 q.

✓ 3. Quais são as duas propriedades fundamentais que uma árvore rubro-negra deve satisfazer antes e depois de qualquer operação? *10/10

- (●) Propriedade da Altura Preta e Propriedade Sem-Borda-Vermelha. ✓
- ( ) Propriedade Inclinada para a Esquerda e Propriedade de Nó Raiz Preto.
- ( ) Propriedade de Balanceamento e Propriedade de Ordem dos Elementos.
- ( ) Propriedade de Filhos Pretos e Propriedade de Folhas Vermelhas.
- ( ) Propriedade da Altura e Propriedade do Grau.

✓ 4. Uma árvore rubro-negra é projetada para ser uma simulação eficiente de qual outra estrutura de dados? *10/10

- ( ) Uma Árvore Scapegoat.
- ( ) Uma Treap.
- ( ) Uma árvore AVL.
- (●) Uma Árvore 2-4. ✓
- ( ) Uma Skiplist.

✓ 5. As operações add(x) e remove(x) em uma árvore rubro-negra são executadas em qual tempo no pior caso? *10/10

- ( ) e) O(n log n).
- (●) d) O(log n) no pior caso. ✓
- ( ) a) O(1).
- ( ) b) O(log n) esperado.
- ( ) c) O(n).

✓ 6. Qual é o número de rotações amortizadas realizadas durante uma operação add(x) ou remove(x) em uma árvore rubro-negra? *10/10

- ○ O(log n).
- ○ O(n).
- ○ O(n log n).
- ◉ Constante. ✓
- ○ Zero .

✓ 7. Qual das seguintes afirmações é verdadeira sobre a implementação das árvores rubro-negras? *10/10

- ○ Sua complexidade é apenas teórica e não afeta a prática.
- ○ São notavelmente simples de implementar, exigindo poucos casos a serem considerados.
- ○ São mais simples de implementar que as árvores AVL.
- ◉ Sua manutenção exige uma análise cuidadosa de vários casos e é complexa, tornando a implementação suscetível a bugs. ✓
- ○ A complexidade se refere apenas à teoria, não à prática de codificação.

✓ 8. Durante o método add_fixup(u) na adição de um nó a uma RedBlackTree, em que situação a operação push_black(g) é realizada? *10/10

- ○ Quando o pai de u é vermelho e u é um filho direito.
- ○ Quando u é a raiz da árvore.
- ○ Quando o irmão de u também é vermelho.
- ◉ Quando o avô g é preto e seu filho direito é vermelho, indicando que ambos os filhos de g são vermelhos, simulando uma divisão de nó 2-4. ✓
- ○ Nunca, push_black é usado apenas na remoção.

✓ 9. Após a remoção de um nó w em uma RedBlackTree, o nó u (o filho de w ou o nó que o substitui) pode ter uma cor "duplo-preto" (valor 2). Qual é o objetivo principal do método remove_fixup(u) em relação a esse estado? *10/10

- ○ Garantir que u seja sempre vermelho para evitar violações.
- ○ Inserir um novo nó para preencher o espaço do nó removido.
- ◉ Executar uma série de rotações e recolorações para mover o nó duplo-preto para cima na árvore até que possa ser eliminado. ✓
- ○ Ignorar a cor duplo-preto, pois é um estado temporário sem consequências.
- ○ Converter a árvore em uma árvore 2-4.

✓ 10. A variante da RedBlackTree apresentada no Capítulo 9 satisfaz uma propriedade adicional: "Em qualquer nó u, se u.left for preto, então u.right é preto"9. Qual é o principal motivo para manter esta propriedade? *10/10

- ○ Aumentar a altura da árvore para melhor desempenho de busca.
- ○ Simplificar a análise amortizada das operações.
- ◉ Reduzir o número de casos encontrados ao atualizar a árvore durante as operações add(x) e remove(x). ✓
- ○ Forçar a árvore a ser sempre uma árvore AVL.
- ○ Permite o uso de randomização nas operações de balanceamento.