50+ Red Black Tree MCQs with FREE PDF

1. What are the operations that could be performed in O(logn) time complexity by red-black tree?

a) insertion, deletion, finding predecessor, successor

b) only insertion

c) only finding predecessor, successor

d) for sorting

Answer: insertion, deletion, finding predecessor, successor

2. When to choose Red-Black tree, AVL tree and B-trees?

a) many inserts, many searches and when managing more items respectively

b) many searches, when managing more items respectively and many inserts respectively

c) sorting, sorting and retrieval respectively

d) retrieval, sorting and retrieval respectively

Answer: many inserts, many searches and when managing more items respectively

3. Which of the following is an application of Red-black trees and why?

a) used to store strings efficiently

b) used to store integers efficiently

c) can be used in process schedulers, maps, sets

d) for efficient sorting

Answer: can be used in process schedulers, maps, sets

4. When it would be optimal to prefer Red-black trees over AVL trees?

a) when there are more insertions or deletions

b) when more search is needed

c) when tree must be balanced

d) when log(nodes) time complexity is needed

Answer: when there are more insertions or deletions

5. What is the below pseudo code trying to do, where pt is a node pointer and root pointer?

```
redblack(Node root, Node pt) :
    if (root == NULL)
       return pt

    if (pt.data < root.data)
    {
        root.left  =   redblack(root.left, pt);
        root.left.parent = root
    }
    else if (pt.data > root.data)
    {
        root.right = redblackt(root.right, pt)
```

```
            root.right.parent = root
        }
    return root
```

a) insert a new node

b) delete a node

c) search a node

d) count the number of nodes

Answer: insert a new node

6. Why Red-black trees are preferred over hash tables though hash tables have constant time complexity?

a) no they are not preferred

b) because of resizing issues of hash table and better ordering in redblack trees

c) because they can be implemented using trees

d) because they are balanced

Answer: because of resizing issues of hash table and better ordering in redblack trees

7. How can you save memory when storing color information in Red-Black tree?

a) using least significant bit of one of the pointers in the node for color information

b) using another array with colors of each node

c) storing color information in the node structure

d) using negative and positive numbering

Answer: using least significant bit of one of the pointers in the node for color information

8. What is the special property of red-black trees and what root should always be?

a) a color which is either red or black and root should always be black color only

b) height of the tree

c) pointer to next node

d) a color which is either green or black

Answer: a color which is either red or black and root should always be black color only

9. Why do we impose restrictions like

. root property is black

. every leaf is black

. children of red node are black

. all leaves have same black

a) to get logarithm time complexity

b) to get linear time complexity

c) to get exponential time complexity

d) to get constant time complexity

Answer: to get logarithm time complexity