Name:_____ Name: _____

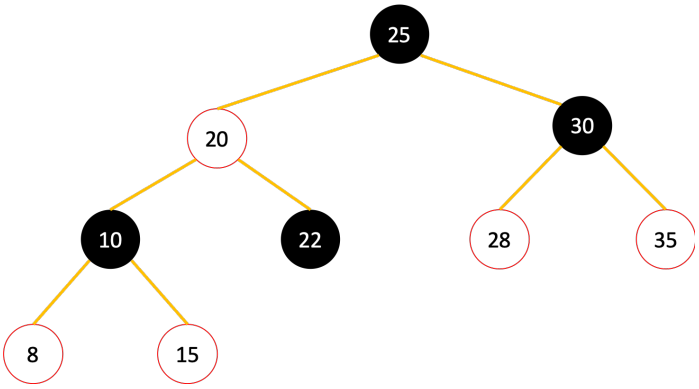
Name:_____ Name: _____

Red-Black Trees

1. What is the minimum number of nodes in a tree based on “k”?

2. Draw a worst-case (as unbalanced as possible) Red-Black tree that has exactly 3 black nodes on every root-NULL path. (Draw red nodes as open circles and black nodes as double lined circles.)

3. Insert a “9” into the tree below. You do **not** have to show all of your work (step through the provided code). You should redraw the tree in the space to the right. This will require several rotations.



```
FUNCTION RBTreeInsert(tree, new_node)
   # Search for position of new_node
   parent = NONE
   current_node = tree.root
   WHILE current_node != NONE
      parent = current_node
      IF new_node.key < current_node.key
         current_node = current_node.left
      ELSE
         current_node = current_node.right
   new_node.parent = parent

   # Insert as root or left/right child
   IF parent == NONE
      tree.root = new_node
   ELSE IF new_node.key < parent.key
      parent.left = new_node
   ELSE
      parent.right = new_node

   # Initialize the new_node
   new_node.left = NONE
   new_node.right = NONE
   new_node.color = RED

   RBTreeFixColors(tree, new_node)
```

```
FUNCTION RBTreeFixColors(tree, node)
   WHILE node.parent.color == RED
      # Look for aunt/uncle node
      IF node.parent == node.parent.parent.left
         aunt = node.parent.parent.right
         IF aunt.color == RED
            node.parent.color = BLACK
            aunt.color = BLACK
            node.parent.parent.color = RED
            node = node.parent.parent
         ELSE
            IF node == node.parent.right
               node = node.parent
               LeftRotate(tree, node)
            node.parent.color = BLACK
            node.parent.parent.color = RED
            RightRotate(tree, node.parent.parent)
      ELSE
         aunt = node.parent.parent.left
         IF aunt.color == RED
            node.parent.color = BLACK
            aunt.color = BLACK
            node.parent.parent.color = RED
            node = node.parent.parent
         ELSE
            IF node == node.parent.left
               node = node.parent
               RightRotate(tree, node)
            node.parent.color = BLACK
            node.parent.parent.color = RED
            LeftRotate(tree, node.parent.parent)
   tree.root.color = BLACK
```