```
// bilanciamento di un RED-BLACK TREE in seguito all'inserimento di un nodo RED
balanceInsert(TREE t)
    t.color ← RED  // coloro il nodo da inserire di rosso

    // t == nil è la condizione di fino ciclo
    while t ≠ nil do
(0)     TREE p ← t.parent                                          // riferimento al padre
        TREE n ← iif(p ≠ nil, p.parent, nil)                       // riferimento al nonno
        TREE z ← iif(n == nil, nil, iif(n.left, n.right, n.left))  // riferimento allo zio
(1)     if p == nil then
            t.color ← BLACK
            t ← nil  // fine
(2)     else if p.color == BLACK then
            t ← nil  // fine
(3)     else if z.color == RED then
            p.color ← z.color ← BLACK
            n.color ← RED
            t ← n  // passo il problema al nonno
        else
(4a)        if (t == p.right) and (p == n.left) then
                rotateLeft(p)
                t ← p  // passo il problema al padre
(4b)        if (t == p.left) and (p == n.right) then
                rotateRight(p)
                t ← p  // passo il problema al padre
            else
(5a)            if (t == p.left) and (p == n.left) then
                    rotateRight(n)
(5b)            else if (t == p.right) and (p == n.right) then
                    rotateLeft(n)

                p.color ← BLACK
                n.color ← RED
                t ← nil  // fine
```