$T(n) = T(n/(4/3)) + 2cn$

$\log_b 1 = 0$ where $b=4/3$

theta(n)

```
Algorithm sortRBG(L)
1        B := new array(3)
1        for i = 0 to 2 do
1            B[i] := new List
n        while !L.isEmpty() do
n            e := L.remove(L.first())
n            if color(e) = red then
n                    B[0].insertLast(e)
n            else if color(e) = blue the
n                    B[1].insertLast(e)
n            else B[2].insertLast(e)
1        for i = 0 to 2 do
n            while ! B[i].isEmpty() do
n                    e := B[i].remove(B[i].first())
n                    L.insertLast(e)
```

```
Algorithm sortRBG(L)
1        B := new Priority Queue
n        while !L.isEmpty() do
n            e := L.remove(L.first())
n            if color(e) = red then
nlogn               B.insertItem(1, e)
n            else if color(e) = blue the
nlogn               B.insertItem(2, e)
nlogn        else B. insertItem(3, e)
n        while ! B.isEmpty() do
nlogn               e := B.removeMin()
n                   L.insertLast(e)
```

```
Algorithm sortRBG(L)
    nextRed := L.first()
    while !L.isLast(nextRed) /\ nextRed.element().color() = Red do
        nextRed := L.after(nextRed)
    curr := nextRed
    while !L.isLast(curr) do
        curr := L.after(curr)
        if curr.element().color() = Red then
            L.swapElements(nextRed, curr)
            nextRed := L.after(nextRed)

Algorithm sortRBG(L)
    next := partitionHelper(L, L.first(), Red)
    partitionHelper(L, next, Blue)

Algorithm partitionHelper(L, next, color)
    while !L.isLast(next) /\ next.element().color() = color do
        next := L.after(next)
    curr := next
    while !L.isLast(curr) do
        curr := L.after(curr)
        if curr.element().color() = color then
            L.swapElements(next, curr)
            next := L.after(next)
    return next
```