# C 4.13

areSame(S1, S2)

for i=0 to S1.size() do O(n)

index <- S2.indexOf(S1[i]) O(n)

if index = -1 then O(n)

return false O(n)

S2.remove(index) O(n)

if !S2.isEmpty() O(1)

return false O(1)

return true O(1)

# R 5.4

1. T(n) = 2T(n/2) + log n

Log2 2 = 1 so case 1 => T(n) is O(n)

1. T(n) = 8T(n/2) + n2

Log2 8 = 3 so case 1 => T(n) is O(n3)

1. T(n) = 16T(n/2) + (nlogn)4

Log2 16 = 4 so case 2 => T(n) is O(n4log5n)

1. T(n) = 7T(n/3) + n

Log3 7 = 1.77 so case 1 => T(n) is O(n1.77)

1. T(n) = 9T(n/3) + (n3log n)

Log3 9 = 2 so case 3 => T(n) is O(n3log n)

a f(n/b) <= δ f(n) for δ <1

9 (n/3)3 log (n/3) = (1/3)n3 log n – log 3 <= δ(n3log n) for δ = 1/3