1) 1) Big(O) -> Upper Bound

a) 2) Big(Ω) -> lower Bound

3) Big(Θ) (theta) -> Avg Bound

$$1 < \log n < n < n\log n < n^2 < 2^n < n!$$

Big (O) Upper Bound

$f(n) = g(n)$ iff $\exists$ +ve c & n_0 such that $f(n) \le c * g(n)$ $\forall n \ge n_0$

$f(n) = 2n + 1$

$2n+1 \le 3n+3$

$2n+1 \le 5n^2$

$2n+1 \le 2n \log n$

$2n+1 \le 2^n$

$2n+1 \le n!$

$f(n) = O(n)$
$f(n) = O(n^2)$
$f(n) = O(n \log n)$
$f(n) = O(2^n)$
$f(n) = O(n!)$

Big Ω Lower Bound

$f(n) = \Omega(g(n))$ iff $\exists$ +ve c and n_0 such that

$f(n) \ge c * g(n)$ $\forall n \ge n_0$

Eg: $2n+1$

$2n+1 \ge 1 \times n$

$2n+1 \ge 1 \times \log n$

$2n+1 \ge 1$

$f(n) = \Omega(n)$
$f(n) = \Omega(\log n)$
$f(n) = \Omega(1)$

Bis (Θ) (Avg Bound)

$f(n) = \Theta(g(n))$ iff $\exists$ +Ve constant c_1, c_2 and n_0 such that $c_1 * g(n) \leq f(n) \leq c_2 * g(n)$

$$f(n) = 2n+1$$

$$\underbrace{1}_{c_1} \times \underbrace{n}_{g(n)} \leq \underbrace{2n+1}_{f(n)} \leq \underbrace{5}_{c_1} \times \underbrace{n}_{g(n)}$$

$\boxed{f(n) = \Theta(n)}$

b) According to definition of Big O, $T(f(n) + g(n)) = O(\max\{f(n), g(n)\})$ if and only if $\exists$ +ve c and n_0 such that $n \geq n_0$

$T(f(n) + g(n)) \leq c \cdot O(\max\{f(n), g(n)\})$

$f(n) \leq \max\{f(n), g(n)\}, \forall n \geq 1$

$g(n) \leq \max\{f(n), g(n)\}, \forall n \geq 1$

Thus, let $c = \cancel{2}$, $n_0 = 1$ we get that for all $n \geq 1$,

$T(f(n) + g(n)) \leq c \cdot O(\max\{f(n), g(n)\})$....

We have two loop invariants

2) ~~We use loop invariants~~

Outer loop: Elements 1 through i-1 are sorted in increasing order.

Inner loop: At begining of jth iteration, minIndex is the index of the smallest element in the range [i, j-1].

Outer loop:

Initialization: At the beginning in the 1st iteration, there not a lot of elements in the range [1,0], so they are sorted in increasing order. The second clause is also true.

Maintenance:

Suppose the invariant holds before iteration i. We show that it holds before iteration i+1 ~~We show that it holds~~. If the inner loop invariants holds, then at the end of the inner loop, minIndex is the index of the smallest element in the array range [i, A.length]. Then the swap ensures that the smallest element in the array in the range [i, A.length] gets placed in A[i]. By previous invocation of the loop invariant, we know that A[1] ≤ A[2] ≤ .... ≤ A[i-1] and A[i] is atleast as large as A[i-1]. ~~Furthermore~~

Termination

At the begining of the (A.length +1)st iteration, elements 1 through A.length are sorted in increasing order, array is sorted.

Innerloop

Initilization: At the begining of the iteration $i+1$, minIndex is i which is the index of smallest element in the range $[i, i]$

Maintanence:

Suppose the invariant holds prior to iteration j. We show that it holds prior to iteration $j+1$. At the begining of the j^{th} iteration, minIndex is the index of the smallest element in the range $[i, j-1]$. If $A[j] < A[minIndex]$, then j is the index of the smallest element in the range $[i, j]$, so setting minIndex $= j$ is correct. If $A[j]$ is not less than $A[minIndex]$ then minIndex is the index of the smallest element in the range $[i, j]$, so it is correct to leave minIndex alone.

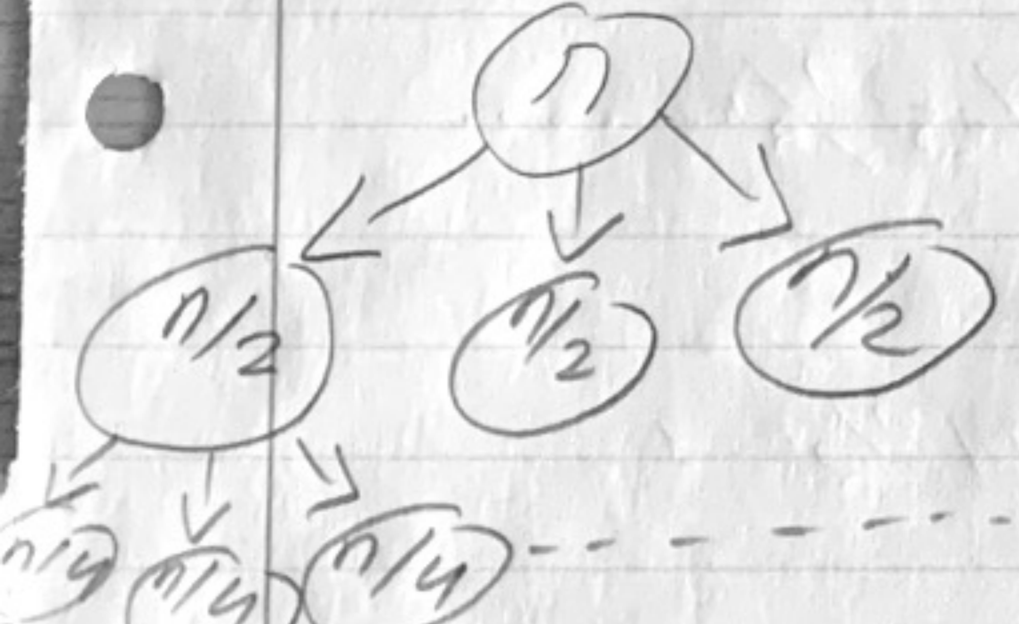
Termination:

At the begining of (A.length+1)st iteration, minIndex is the index of smallest element in the range $[i, A.length]$, which is the condition we needed for our proof of the outer loop invariant.

3) $T(n) = \underbrace{3T(n/2)}_{\text{rec}} + \underbrace{2n}_{\text{non-rec}}$

n Size	Work of one call	Calls at this layer	total non-recursive work at layer
n	$2n$	1	$2n$
$n/2$	$2 \cdot \frac{n}{2}$	3	$3n/2$
$n/4$	$2 \cdot n/4$	9	$9n/2$
$n/8$	$2 \cdot n/8$	$9 \cdot 3 = 27$	$27n/4$

1	$2 \cdot 1 = O(1)$	$3^{\log_2 n}$	$3^{\log_2 n}$



How many times did n get cut in half?

$K = \log_2 n$

$2n, 3n, 9n/2, 27n/2$

$\times 3/2 \quad \times 3/2 \quad \times 3n/2$

$S_n = \dfrac{a(r^k - 1)}{r - 1}$ $\quad r = 3/2$ $\quad a = 2n$ $\quad K = \log_2 n$

$$\frac{2n\left((3/2)^{\log_2 n} - 1\right)}{3/2 - 1} = 4n\left(\left(3/2\right)^{\log_2 n} - 1\right)$$

$$T(n) = 4n\left(\tfrac{3}{2}\right)^{\log_2 n} - 4n = 4n\,\frac{3^{\log_2 n}}{2^{\log_2 n}} - 4n$$

$$= \cancel{4n}\,\frac{3^{\log_2 n}}{\cancel{4}} - \cancel{4n}$$

$$= O\left(3^{\log_2 n}\right)$$

$$= O\left(n^{1.6}\right)$$

4) $4T(n/2) + n^2 \log n$

$$
\begin{aligned}
T(n) &= 4T(n/2) + n^2 \log n \\
&= 4[4T(n/2^2) + (n/2)^2 \log n/2] + n^2 \log n \\
&= 4^2 T(n/2^2) + 4(n/2)^2 \log n/2 + n^2 \log n \\
&= 4^2[4T(n/2^3) + (n/2^2)^2 \log n/2^2] 4(n/2)^2 \log n/2 + \\
&\quad n^2 \log n \\
&= 4^3 T(n/2^3) + 4^2(n/2^2)^2 \log n/2^2 + 4(n/2)^2 \log n/2 + \\
&\quad n^2 \log n \\
&\ \vdots \\
&= 4^k T(n/2^k) + 4^{k-1}(n/2^{k-1})^2 \log n/2^{k-1} \dots \\
&\quad 4(n/2)^2 \log n/2 + n^2 \log n
\end{aligned}
$$

This will continue until $n/2^k = 1$,

$k = \log_2 n$

$4^2(n/2^2)^2 \log(n/2)^2 \; 4(n/2)^2 \log(n/2) + n^2 \log n$

$n^2 \log(n/2)^2$

$n^2(\log(n/2)^2 + \log(n/2) + \log n$

$T(n) = n^2 \log((n/2)^2 \times (n/2) \times n)$

4) b- Base Case:

$n=1$

$T(1) = c + 1^2 / \log^2 1 = c$

$= O(1^2/\log 1)\ C$

$0 \leq C$

$T(k)$ is true

Then $T(k) \leq k^2 C, k^2 / \log^2 k = O(k^2/\log^2 k)$

$T(k+1) = O((k+1)^2 / \log^2 (k+1))$

$(k+1)^2 c + (k+1)^2 / \log^2 (k+1) = O((n+1)^2 / \log^2 (n+1))$

Therefore, $T(n) \leq n^2 c + n^2 / \log^2 n$;

Proof by induction: $T(n) = O(n^2/\log^2 n)$

5) $T(n) = a \cdot T(n/b) + O(n^d)$

a) $T(n) = 3T(n/4) + \sqrt{n}$

$a = 3$

$b = 4$

$d = 1/2$

$a > b^d$

$3 > 4^{1/2}$

$3 > 2$

$O(n^{\log_4 3})$

b) $T(n) = 7T(n/3) + O(n^3)$

$a = 7$

$b = 3$ $\quad a < b^d$

$d = 3$ $\quad 7 < 9$

$O(n^3)$

c) $T(n) = 2T(n/3) + n^c$

$a = 2$

$b = 3$ $\quad 2 < 3$

$d = c$ $\quad O(n^c)$