(2) Small Oh (o) notation: We denote o notation to denote an upper bound that is not asymptotically tight.

$o(g(n)) = \{f(n)$: for any positive constant $c > 0$ there exists a constant $n_0 > 0$ such that

$0 \leq f(n) < g(n) \ \forall \ n \geq n_0$

(3) Big Omega ($\Omega$) notation.

$\Omega(g(n)) = \{f(n)$: for any positive constant $c$ and $n_0\}$

$0 \leq g(n) \leq f(n) \ \forall \ n \geq n_0$

(4) Small omega ($\omega$) notation

(5) Theta notation ($\theta$)

Q2. $O(\log n)$

Q3. $T(n) = \begin{cases} 3(T(n-1)) & n > 0 \\ 1 & n \le 0 \end{cases}$

By using back substitution.

$T(n) = 3T(n-1)$ — (1)

$T(n-1) = 3T(n-2)$ — (2)

$T(n-2) = 3T(n-3)$ — (3)

Put ~~3~~ 2 & 3 in (1)

$T(n) = 3.3.3T(n-3)$

$T(n) = 3^k T(n-k)$

let $k = n$

$T(n) = 3^n T(n-n)$

$= 3^n$

$= O(3^n)$

Q4) $T(n) = \begin{cases} 2(T(n-1)) - 1 & n > 0 \\ 1 & n < 0 \end{cases}$

Using back substitution

$T(n) = 2T(n-1) - 1$

$T(n-1) = 2(T(n-2)) - 1$

$T(n-2) = 2T(n-3) - 1$

$T(n) = 2 \cdot 2T(n-2) - 2 - 1$

$T(n) = 2 \cdot 2 \cdot 2T(n-3) - 4 - 2 - 1$

$T(n) = 2^k T(n-k) - 2^{k-1} - 2^{k-2} \dots 1$

let $k = n$

$T(n) = 2^n T(n-n)$

$- [1 + 2 + 4 + \dots + 2^{k-1}]$

$= 2^n T(1) - [1(2^{k-1} - 1)]$

$= 2^n - 2^{n-1} - 1$

$= O(2^n)$

Q5. $O(n)$

Q6. $O(\sqrt{n})$

Q7. $O(n \log n \log n)$

Q8. The recurrence relation is

$T(n) = T(n-3) + n^2 \quad n > 1 \quad \text{—(1)}$
$\qquad\qquad\qquad\qquad\qquad n <= 1$

By back substitution.

$T(n) = T(n-6) + (n-3)^2$

$T(n-6) = T(n-9) + (n-6)^2$

Put back in eq (1)

$T(n) = T(n-9) + (n-6)^2 + (n-3)^2 + n^2$

$T(n) = T(n-3k) + (n-3(k-1))^2 + (n-3(k-2))^2 + \ldots n^2$

Let $3k = n$

$k = n/3$ —(2)

$$T(n) = T\left(n - 3 \times \frac{n}{3}\right) + \underbrace{n^2 + (n-3)^2 + \ldots + \left(n - 3\left(\frac{n}{3} - 1\right)\right)^2}_{\text{total of } K \text{ terms}}$$

$$T(n) = T(1) + n^2 + (n-3)^2 + (n-6)^2 + \ldots + (n - n + 3)^2$$

$$T(n) = 1 + (n^2 + n^2 + n + \ldots) + (\underbrace{xn + yn + zn +}_{\text{can be ignored}})$$

$$T(n) = 1 + kn^2 + \ldots$$

$$T(n) = 1 + \frac{n^3}{3}$$

$$= O(n^3).$$

Q9 Sum loop runs as

$n + n/2 + n/3 + \cdots \; n/n$

$\underbrace{n\left(1 + 1/2 + 1/3 + \cdots 1/n\right)}$

This sum will converge to $\log n$.

Hence $n(\log n)$

$\Rightarrow O(n \log n)$

Q10 $f(n) = n^k$. $\quad k >= 1$

$g(n) = a^n$. $\quad a > 1$

Exponential function grow faster than polynomial functions hence.

$O(n^k) < O(a^n)$

for values of $k \geq x$ and $a \geq y$.

Calculate $x$ & $y$

Let $k = 2$ and $a = 2$ as well

$f(n) = n^2$, $g(n) = 2^n$.

take log on both sides

$\log(f(n)) = 2\log_2(n)$

$\log(g(n)) = n\log_2 2$

$O(\log n) < O(n)$

Hence for $k >= 2$ and $a >= 2$ the condition ~~sta~~ satisfies.

Q11 $O(\sqrt{n})$, because the values of i go as follows: 1, 3, 6, 10, 15, 2

Also we know that

$$f(n) = \frac{n(n+1)}{2}$$

for the sum of series

$1+2+3+4 \ldots$

So the series 1, 3, 6, 10, 15

will stop when $a_n$

becomes equal to a greater than n

$$\because \frac{n(n+1)}{2} = n_0$$

$$n = \sqrt{n_0}$$

Q12) $T(n) = \begin{cases} T(n-1) + T(n-2) + 1 & n \geq 2 \\ 1 & 0 \leq n < 2 \end{cases}$

assume time taken by $T(n-2)$
$\simeq T(n-1)$

So $T(n) = 2T(n-2) + c$ [ c is a constant ]

solving we get :

$$T(n) = 2.2.2T(n-2.3) + 3C + 2C + 1C$$

$$T(n) = 2^k T(n-2k) + (2^k - 1)c$$

$$n - 2k = 0 \qquad k = n/2$$

$$T(n) = 2^{n/2} T(0) + (2^{n/2} - 1)c$$

$$T(n) = O(2^{n/2}) \simeq O(2^n)$$

(14) $T(n) = T(n/4) + T(n/2) + Cn^2$

We can assume $T(n/2) \geq T(n/4)$

$T(n) = 2T(n/2) + Cn^2$

using Masters theorem

$a = 2$, $b = 2$

$\log_b a = \log_2 2 = 1$

$f(n) = Cn^2$

$n^k = n^2$

$k = 2$

$\log_b a < k$

$1 < 2$

$+ Cn^2$

)

Complexity $= O(n^k)$
$= O(n^2)$

(15) Inner loop will run $1/i$ times

$= 1 + 1/2 + 1/3 + 1/4 + \cdots 1/n$

$= n(1 + 1/2 + 1/3 + \cdots 1/n)$

$= n \log n$

$= O(n \log n)$

Q18 (a) $100 < \log(\log(n)) < \log(n)$
$< \sqrt{n} < n < \log(n!) < n \log n < n^2$
$< 2^n < 2^{2n} < 4^n < n!$

(b) $1 < n < 2n < 4n < \log(\log n) < \log(\sqrt{n})$
$< \log(n) < \log(2n) < 2\log(n)$
$< \log(n!) < n\log(n) < n^2 < (2^n)2$
$< n!$

(c) $96 < \log_8(n) < \log_2(n) < n\log_6 n$

$< n\log_2(n) < \log(n!)$

$< 5n < 8n^2 < 7n^3 < 8^{2n} < n!$

Q19 for(int i=0; i<n; i++)

{

if (arr[i] is equal to key)

print index and break

else

continue

}.

Q21) Quicksort : $O(n \log n)$
Merge sort : $O(n \log n)$
Bubble sort : $O(n^2)$
Selection sort : $O(n^2)$
insertion sort : $O(n^2)$

22) Inplace : Bubblesort, Selection sort, Quicksort, Insertionsort

Stable : Bubblesort, Insertion sort, Mergesort.

Online : Insertion sort.

24) Recurrence relation for Binary search :

$$T(n) = T(n/2) + 1$$