Ques 1 –

Asymptotic notations are the mathematical notations used to describe the running time of a algorithm, when the input tends towards a particular value or a limiting value.

For Example: In Bubble sort, when the given array is already sorted i.e best case

(i) Big O notation (O): (Asymptotic upper bound) The function $f(n) = O(g(n))$ iff there exist a positive constant C such that $f(n) \leq C * g(n)$

ii) Big Omega notation (Ω): (Asymptotic lower bound) the function $f(n) = \Omega(g(n))$ iff there exist a positive constant C such that $f(n) \geq C * g(n)$

(iii) Big Theta notation (Θ): (Asymptotic tight bound) The function $f(n) = \Theta(g(n))$, iff there exist a positive constants C_1, C_2, K such that $C_1 * g(n) \leq f(n) \leq C_2 * g(n)$ for $n \geq k$.

(iv) Small o notation: Let $f(n)$ and $g(n)$ be functions such that $f(n) = o(g(n))$ iff there exist C constant such that $f(n) < C * g(n)$
$n > K \Rightarrow C > 0$.

(v) Small omega notion (ω): Let $f(n)$ and $g(n)$ be function such that $f(n) = \omega(g(n))$ iff there exist C constant such that $f(n) > C * g(n)$
$n > K \Rightarrow C > 0$.

Ques 2 →

```
for (i=1 to n)
{
  i = i*2;
}
```

$i = 1, 2, 4, 8 \ldots n$

$a = 1 \quad r = 2$

n^{th} term $= 2^k$

Assume $i >= n$ (terminating condition)

$\therefore i > 2^k$

$2^k >= n$ taking log

$\log k = \log_2 n$

$\boxed{O(\log_2 n)}$

Ques 3 –

$T(n) = \{ 3T(n-1) \text{ if } n > 0, \text{otherwise } 1 \}$

$$T(n) = \begin{cases} 1 & n = 0 \\ 3T(n-1) & n > 0 \quad \text{—①} \end{cases}$$

backward substitution

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

$3T(n-2)$ —②

put ② in ①

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

$T(n) = 9T(n-2)$ —③

$T(n-2) = 3T(n-2-1) = 3T(n-3)$ —④

③ in ④

$T(n-3) = 9(3T(n-3)) = 3^3 T(n-3)$

$T(n) = 27T(n-3)$

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

$n - k = 1 \quad k = n-1$

$T(n) = 3^{n-1} T(1)$

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

Ques4 → $T(n) = 2T(n-1) - 1$ — ①

$T(1) = 1$

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

$T(n-1) = 2T(n-2) - 1$ — ②

→ $T(n) = 2(2T(n-2) - 1) - 1$ ② in ①

$T(n) = 4T(n-2) - 3$ — ③

$T(n-2) = 2T(n-3) - 1$ — ④

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

$8T(n-3) - 7$ — ⑤

For any value k

$T(n) = 2^k T(n-k) - (2^k - 1)$ — ⑤

let $n-k=1$

$n = k$

$k = n-1$

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

$2^{n-1} - 2^{n-1} + 1 = 1$ $\boxed{T(n) = Q1}$

Ques5 →

```
int i=1, s=1;
while (s<=n)
{
  i++;
  s=s+i;
  printf("#");
}
```

i	s
1	1
2	1+2
3	1+2+3
.	.
.	.
k	1+2+3+...+k

$1 + 2 + 3 + 4 + \dots + k \leq n$

$\frac{k(k+1)}{2} \leq n$

$k^2 \leq n$

$k = \sqrt{n}$

$\therefore O(\sqrt{n})$

Ques 5 → void function (int n)

```
{ int i, j, k, count = 0;
  for (i=n/2 ; i<=n ; i++)          —— ① n
    for (j=1 ; j<=n ; j=j*2)        —— ② n log n
      for (k=1 ; k<=n ; k=k*2)      —— ③ (n+1) log²n
        count++;
}
```

for the loop 1, the time complexity is $\log n$

for the loop 2, the time complexity is $(n+1)\log n$

for the loop 3, the time complexity is $n \cdot \log^2 n$

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

Ques 6 →

```
void function (int n) {
int i, count = 0;
  for (i=1 ; i*i <= n ; i++)
      count++;
}
```

$i * i > n$ is termination condition

$k^2 \geq n$

$k = \sqrt{n}$

$\therefore O(\sqrt{n})$

Ques 8 →

```
function (int n) {
  if (n == 1) return;
  for (i=1 to n) {          —— n ①
    for (j=1 to n) {        —— n*(n-1) ②
      Printf ("*");
    }
  }
  function (n-3);
}
```

for loop ① the code run n times

For the loop ② the code run $n*(n-1)$ times

$O(n^2)$

Ques 9 →

```
void function (int n) {
for (i=1 to n) {
 for (i=1 to n) {          —— ①
  for (j=1; j<=0; j=j+i)   —— ②
   Printf ("*")
 }
}
```

For loop 1 the time complexity is n

For loop 2 the time complexity $(n-1) * \log n$

$O(n \log n)$

ves 10? for the function n^k and C^n, what is asymptotic relationship b/w these function? Assume that $k>=1$ and $C>1$ are constants. find out value of C and no for which relation hold!

The asymptotic relation between n^k & C^n is $\boxed{n^k = O(C^n)}$

i.e $n^k \leq C_1 \cdot (C^n)$

$\Rightarrow n^k = C_1 . C^n$

Put $n=2$ $k=2$ & $C=2$

$(2)^2 = C_1 . (2)^2$

$4 = C_1 . 4$

$\boxed{C_1 = 1}$