Excercise

Q-1 What is the Time Complexity of following functions?

a.

```
void fun(int n, int arr[])
{
   int i = 0, j = 0;
   for (; i < n; ++i)
      while (j < n && arr[i] < arr[j])
         j++;
}
```

b.

```
void fun(int n, int arr[])
{
   int i = 0, j = 0;
   for (; i < n; ++i) {
      j = 0;
      while (j < n && arr[i] < arr[j])
         j++;
   }
}
```

Q-2 In a competition, four different functions are observed. All the functions use a single for loop and within the for loop, same set of statements are executed. Consider the following for loops:

```
A) for(i = 0; i < n; i++)

B) for(i = 0; i < n; i += 2)

C) for(i = 1; i < n; i *= 2)
```

```
D) for(i = n; i <= n; i /= 2)
```

Q-3 Consider the following functions:

$f(n) = 2^n$

$g(n) = n!$

$h(n) = n^{\log(n)}$

Which of the following statements about the asymptotic behavior of f(n), g(n), and h(n) is true?

(A) f(n) = O(g(n)); g(n) = O(h(n))

(B) f(n) = *Omega* (g(n)); g(n) = O(h(n))

(C) g(n) = O(f(n)); h(n) = O(f(n))

(D) h(n) = O(f(n)); g(n) = *Omega* (f(n))

Q-4 In the following C function, let n >= m.

```
int gcd(n, m)
{
   if (n % m == 0)
      return m;
   n = n % m;
   return gcd(m, n);
}
```

How many recursive calls are made by this function?

(A) *theta* (log(n))

(B) *Omega* (n)

(C) *theta* (log(log(n)))

(D) *theta* (sqrt(n))

Q-5 Consider the following functions

- f(n) = 3n^{\sqrt{n}}
- g(n) = 2^{\sqrt{n}{\log_{2}n}}
- h(n) = n!

Which of the following is true? (GATE)

(A) h(n) is 0(f(n))

(B) h(n) is 0(g(n))

(C) g(n) is not 0(f(n))

(D) f(n) is 0(g(n))

Q-6 Let s be a sorted array of n integers. Let t(n) denote the time taken for the most efficient algorithm to determine if there are two elements with sum less than 1000 in s. which of the following statements is true? (GATE)

a) t (n) is 0 (1)

b) n < t (n) < n log2n *log2n*

c) n log 2 n < t (n) < n^2

d) t (n) = n^2

Q-7 The recurrence equation

T(1) = 1

T(n) = 2T(n - 1) + n, n ≥ 2

evaluates to

a. 2^{n+1}- n - 2

b. 2^n - n

c. 2^{n+1} - 2n - 2

d. 2^n + n

Q-8 Consider the following three claims

1. $(n + k)^m = \Theta(n^m)$, where k and m are constants

2. $2^{n+1} = O(2^n)$

3. $2^{2n+1} = O(2^n)$

Q-9

Let T(n) be a function defined by the recurrence $T(n) = 2T(n/2) + \sqrt{n}$ for $n \geq 2$ and $T(1) = 1$

Which of the following statements is TRUE?

$T(n) = \theta(\log n)$

$T(n) = \theta(\sqrt{n})$

$T(n) = \theta(n)$

$T(n) = \theta(n \log n)$

Q-10 Arrange the following functions in increasing asymptotic order:

A. $n^{1/3}$

B. e^n

C. $n^{7/4}$

D. $n \log^9 n$

E. 1.0000001^n

Q-11 If T1 = O(1), give the correct matching for the following pairs:

(M) Tn=Tn−1+n	(U) Tn=O(n)
(N) Tn=Tn/2 +n	(V) Tn=O(nlogn)
(O) Tn=Tn−1 +logn	(W) T=O(n^2)

Q-12 Give the correct matching for the following pairs:

A. O(log n)	1. Selection sort
B. O(n)	2. Insertion sort
C. O(nlog n)	3. Binary search
D. O(n^2)	4. Merge sort

Q-13 Which of the following is False?

a. $100n \log n = O(\frac{n \log n}{100})$

b. $\sqrt{\log n} = O(\log \log n)$

c. If $0 < x < y$ then $n^x = O\left(n^y\right)$

d. $2^n \neq O\left(nk\right)$

Q-14 Given A, an array of size n, comprised of an increasing sequence of numbers followed immediately by a decreasing one. What is worst case time complexity of optimal algorithm to determine if a given number x is in the array?

o Θ(log n)

o Θ(n)

o Θ(n^2)

o Θ(log n)^2