1. What is time complexity of fun()?

```
def fun(n):
    count = 0
    i = n
    while i > 0:
        for j in range(i):
            count += 1
        i //= 2
    return count
```

(A) $O(n^2)$

(B) O(n*log(n))

(C) O(n)

(D) O(n*log(n*Log(n)))

2. What is the time complexity of fun()?

```
def fun(n):
    count = 0
    for i in range(n):
        for j in range(i, 0, -1):
            count += 1
    return count
```

(A) Theta (n)

(B) Theta (n^2)

(C) Theta (n*log(n))

(D) Theta (n*(log(n*log(n))))

3. $O(n^2)$ is the worst case time complexity, so among the given options it can represent :

(A) O(n)

(B) O(1)

(C) O (nlogn)

(D) All of the above

4. Which of the given options provides the increasing order of asymptotic complexity of functions f1, f2, f3, and f4?

$f1(n) = 2^n$
$f2(n) = n^{(3/2)}$
$f3(n) = n*log(n)$
$f4(n) = n^{log(n)}$

(A) f3, f2, f4, f1

(B) f3, f2, f1, f4

(C) f2, f3, f1, f4

(D) f2, f3, f4, f1

5. What is the time complexity of the below function?

```
def fun(n, arr):
    i = 0
    j = 0
    while i < n:
        while j < n and arr[i] < arr[j]:
            j += 1
        i += 1
```

(A) O(n)

(B) $O(n^2)$

(C) O(n*log(n))

(D) $O(n*log(n)^2)$

6. What does it mean when we say that an algorithm X is asymptotically more efficient than Y?

(A) X will be a better choice for all inputs

(B) X will be a better choice for all inputs except possibly small inputs

(C) X will be a better choice for all inputs except possibly large inputs

(D) Y will be a better choice for small inputs

7. What is the time complexity of the function?

```
def unknown(n):
    k = 0
    i = n // 2
    while i <= n:
        j = 2
        while j <= n:
            k = k + n // 2
            j = j * 2
        i += 1
    return k
```

(A) n^2

(B) n logn

(C) n^3

(D) n^3 logn

8. Time Complexity of the following code?

```
int Sum1ToN(int n) {
  int ans = n*(n+1)/2;
  return ans;
}
```

- ○ O(N)
- ○ O(1)
- ○ O(N*N)
- ○ O(LogN)

9. what is the time complexity of the following code?

```
int getSum(int n, int m) {
  int ans = 0;
  for(int i = 1; i <= n; ++i)
    for(int j = 1; j <= m; ++j)
      ans++;
  return ans;
}
```

- ○ O(N)
- ○ O(M)
- ○ O(N + M)
- ○ O(N*M)

10. What is the time complexity of the following code?

```
int doRandomStuff(int n, int m) {
  int ans = 0;
  for(int i = 1; i <= n; ++i) {
    int var = n;
    while(var > 0) {
        // do some O(1) operation.
        var /= 2;
    }
  }

  for(int j = 1; j <= m; ++j)
    ans++;
  return ans;
}
```

- ○ O(N*LogN)
- ○ O(N + LogN + M)
- ○ O(N + M)
- ○ O(N*LogN + M)

11. What is the time complexity?

```
def fun(n):
    i = 1
    while i <= n:
        i *= 2
```

A) O(n)
B) O(log n)
C) O(n log n)
D) O(1)

12. Time complexity?

```
def fun(n):
    for i in range(n):
        for j in range(1, n, 2):
            print(i, j)
```

A) O(n)
B) O(n log n)
C) O(n²)
D) O(n² / 2)

13. Time complexity?

```
def fun(n):
    for i in range(n):
        j = i
        while j > 0:
            j //= 2
```

A) O(n)
B) O(n log n)
C) O(n²)
D) O(log n)

14.

```
def fun(n):
    for i in range(n):
        for j in range(i):
            print(i, j)
```

Time complexity?

A) O(n)
B) O(n log n)
C) O(n²)
D) O(n³)

15. def fun(n):
    i = n
    while i > 0:
        for j in range(i):
            print(j)
        i //= 2
    Time complexity?

    A) O(n)
    B) O(log n)
    C) O(n log n)
    D) O(n²)

16. def fun(n):
    for i in range(n):
        for j in range(n):
            break
    Time complexity?

    A) O(n²)
    B) O(n)
    C) O(log n)
    D) O(1)
17. def fun(n):
    return n * (n + 1) // 2
    Time complexity?

    A) O(n)
    B) O(log n)
    C) O(1)
    D) O(n log n)

18. def fun(n):

```
    i = 1
    while i < n:
        for j in range(n):
            print(j)
        i *= 2
```

Time complexity?

A) O(n)
B) O(n log n)
C) O(n²)
D) O(log n)

19.
```
def fun(arr):
    max_val = arr[0]
    for x in arr:
        if x > max_val:
            max_val = x
```

Time complexity?

A) O(1)
B) O(log n)
C) O(n)
D) O(n²)

20.
```
def fun(n):
    i = 0
    while i < n:
        j = 0
        while j < n:
            j += 1
        i += 1
```

Time complexity?

A) O(n)
B) O(n log n)
C) O(n²)
D) O(log n)