Design and Analysis of Algorithms - Assignment 2, Due: 17/Sep.
Online Submission, 18/Sep, 2-2.30 PM: The link will be shared on 18/Sep, 2 PM. You are expected to discuss as a group and keep the solution ready before the time line.

Each question carries 1 mark. For all questions, if logarithm appears, then it is log base 2.

1. The solution to the recurrence $T(n) = T(n-1) + 2n - 1$.
   (a) n^2
   (b) $n^2 - n$
   (c) $n^2 + n$
   (d) none of the above
2. Tick the correct one.
   (a) All recurrence relations can be solved using Master theorem.
   (b) Step count analysis gives a precise estimate of the time complexity even if there are selection statements.
   (c) The asymptotic analysis focuses on the order of growth, focusing on the contribution of leading term in the polynomial.
   (d) All asymptotic functions are comparable using asymptotic notation.
3. Tick the correct one.
   (a) For every iterative algorithm, there is an equivalent (no change in the functionality) recursive algorithm. Converse is false.
   (b) For every recursive algorithm, there is an equivalent iterative algorithm. Converse is false.
   (c) Recursive algorithm if and only if iterative algorithm, for any computational problem.
4. Which of the following statement is false.
   (a) Iterative algorithm for Fibonacci sequence runs in polynomial time whereas the recursive algorithm is exponential
   (b) Iterative algorithm for Factorial runs in polynomial time whereas the recursive algorithm is exponential
   (c) Listing all permutations of a set of size n incurs exponential effort.
   (d) List all k size subsets of a set of size n incurs polynomial effort if k is fixed.
5. Which one of the following has an exponential time algorithm and no polynomial time algorithm.
   (a) List all subsets of a set of size 10.
   (b) Finding duplicates in an integer array of size n.
   (c) Printing the first 1000 numbers of a Fibonacci sequence
   (d) Identifying a subarray in an integer array whose sum is a given integer.
6. What is the time complexity of the following code.

```
main( )
{ printf("enter an integer");
scanf("%d",&a); }
```

   (a) $\Theta(1), O(1), o(\log n)$

(b) $o(1), \omega(1), O(1)$

(c) $\Omega(1), o(1), O(\log n)$

(d) $\Theta(1), O(\log n), \omega(1)$

7. The recurrence that represents the listing of all permutations of a set of size n

   (a) $T(n) = nT(n-1) + n$

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

   (c) $T(n) = T(n-1) + n!$

   (d) $T(n) = T(n-1) + 2^n$

8. A problem has six different algorithms $\log \sqrt{n}, 4^{\log_2 n}, n\sqrt{n}, 2^{n \log n}, n^n, \log n$. Arrange them in non-decreasing order of complexity.

   (a) $\log \sqrt{n}, \log n, n\sqrt{n}, n^n, 2^{\log_2 n}, 4^{\log_2 n}$

   (b) $\log \sqrt{n}, \log n, 4^{\log_2 n}, n\sqrt{n}, n^n, 2^{n \log_2 n}$

   (c) $\log \sqrt{n}, \log n, n\sqrt{n}, 4^{\log_2 n}, n^n, 2^{n \log_2 n}$

   (d) $\log \sqrt{n}, n\sqrt{n}, \log n, 4^{\log_2 n}, n^n, 2^{n \log_2 n}$

9. The asymptotic complexity of the step count function $n^2 \log n + 3.001^n \log n + n^{2.1} - 17$

   (a) $\Theta(3^n), \Omega(4^n), \omega(n^{2.1})$

   (b) $\Theta(3.001^n), \Omega(n^{2.1}), o(4^n)$

   (c) $\Omega(n^2 \log n), \omega(n^{2.1}), o(4^n)$

   (d) $O(3^n), \omega(1), \Omega(n^2 \log n)$

10. The time complexity of an algorithm is given by $\{n^2$ if n is even, n, if n is odd $\}$. The asymptotic notation is given by

    (a) $O(n^2), \Omega(n), \Theta(n^2)$

    (b) $O(n^2), \Omega(n)$, Theta cannot be determined.

    (c) $\Omega(n), \Theta(n), o(n^3)$

    (d) $\omega(1), \Omega(n^2)$, Theta cannot be determined.

11. The solution to the recurrence $T(n) = 4T(\frac{n}{2}) + n^2 \log n$,

    (a) Case 3 of Master theorem, $T(n) = \Theta(n^2 \log n \log n)$

    (b) Case 2 of Master theorem, $T(n) = \Theta(n^2 \log n)$

    (c) Case 1 of Master theorem, $T(n) = \Theta(n^{2.1})$

    (d) Master theorem cannot be applied, using recurrence tree, we get $T(n) = \Theta(n^2 \log n \log n)$

12. The solution to the recurrence $T(n) = 2^n T(\frac{n}{6}) + 4^n$,

    (a) Master theorem cannot be applied, solution is $\Theta(n \cdot 2^n)$

    (b) Case 3 of Master theorem, $\Theta(4^n)$

    (c) Master theorem cannot be applied, solution is $\Omega(4^n)$ but not $\Theta(4^n)$

    (d) Case 1 of Master theorem, $\Theta(n^{\log_6 2^n})$

13. The solution to the recurrence $T(n) = T(\sqrt{n} + 5) + 2^n$

(a) $\Theta(\sqrt{n})$

(b) $\Theta(n)$

(c) $\Theta(2^n)$

(d) $\Theta(\sqrt{n}\log n)$

14. What is the complexity of this code

```
int i=1,n; // input size: n
while(i <=n) i=i*3 ;
int k, j=1; // k is a fixed integer
while(j <= n) j=j*k;
```

(a) $\Theta(\log_3 n) + \Theta(\log_k n)$

(b) $O(\log_3 n) + \Omega(\log_k n)$

(c) $\Theta(\log_3 n) + \Omega(\log_k n)$

(d) $\Omega(\log_3 n) + \Omega(\log_k n)$

(a) Only D

(b) Only A,B

(c) Only A

(d) All are true

15. Pick all that are true

(a) $o(g(n)) \cap \omega(g(n)) = \emptyset$

(b) $o(g(n)) \subset O(g(n))$

(c) $o(g(n)) \cap \Theta(g(n)) = \emptyset$

(d) $O(g(n)) \cap \Omega(g(n)) = \Theta(g(n))$

(a) All are true

(b) Only D

(c) Only A,B

(d) Only B,D

16. The solution to the recurrence $T(n) = 2T(\frac{n}{2}) + n \cdot n^{1+sinn}$

(a) Case 2 of Master theorem, $T(n) = \Theta(n \log n)$

(b) Master theorem cannot be applied, however $T(n) = O(n^3)$

(c) Case 3 of Master theorem, $T(n) = \Theta(n \cdot n^{1+sinn})$

(d) Case 3 of Master theorem, however regularity condition fails.

(a) Only C

(b) Only B

(c) Only B,D

(d) Only D

17. Let $T(n) = aT(\frac{n}{b}) + f(n)$, $a, b > 1$. Which of the following represents the asymptotic complexity of $T(n)$.

(a) $\Omega(f(n)), \Omega(n^{\log_b a}), \omega(n^{\log_b a - \epsilon}), \epsilon > 0$

(b) $\Omega(f(n)), \Omega(n^{\log_b a}), \omega(n^{\log_b a})$

(c) $\Theta(f(n)), \Omega(n^{\log_b a}), \omega(n^{\log_b a} + \epsilon), \epsilon > 0$

(d) $\Omega(\log n), \Omega(f(n)), \omega(1)$

(a) Only A,B

(b) Only A,B,C

(c) Only A

(d) Only A,D

18. Consider a problem P and there are two algorithms to solve P. The complexity of A_1 is $100n^2$ and A_2 is $0.0001\ 2^n$.

(a) A_2 is asymptotically faster than A_1 for all inputs whose size greater than or equal to 30.

(b) A_1 is asymptotically faster than A_2 for all input size $n \geq n_0$, for some n_0.

(c) For input size at least 10, A_2 is asymptotically faster than A_1

(d) A_1 and A_2 cannot be compared.

19. The solution to the recurrence $T(n) = T(\frac{n}{2}) + T(\frac{n}{4}) + T(\frac{n}{8}) + n$,

(a) $O(n)$ but not $\Theta(n)$

(b) $\Omega(n)$, $O(n \log n)$, but theta cannot be determined.

(c) $\Theta(n)$

(d) $\Omega(n \log n)$ and Master theorem cannot be applied.

20. For the recurrence relation $T(n) = 3T(\frac{n}{2}) + n$, the number of nodes at level 2 and the number of nodes at the last but one level are

(a) $9, n$

(b) 9, one third of the number of leaves.

(c) $3, 3^{\log n}$

(d) 9, half of the number of leaves.

21. Consider the first pass of the bubble sort algorithm; let x denote the number of comparisons and y denote the number of swaps. For the input $(1, 2, \ldots, \frac{n}{2}, n, n-1, \ldots, \frac{n}{2}+1)$, assume n is even. What is the value of x and y

(a) $\frac{n}{2} - 1, \frac{n}{2} - 1$

(b) $n - 1, \frac{n}{2} - 1$

(c) $n - 1, 0$

(d) None of the above

22. Consider the bubble sort algorithm; let x denote the number of passes to find minimum, y denote the number of passes to find maximum, z denote the number of passes to find second maximum. Then, the value of (x, y, z)

(a) $(n, 1, 2)$

(b) $(n - 1, n, n)$

(c) $(n - 1, 1, 2)$

(d) $(n-1, n-1, n-1)$

23. The complexity of finding k^{th} maximum in an integer array, k is a fixed integer (say, $5, 10$).

    (a) Insertion sort takes $O(n^2)$ whereas bubble sort takes $O(n)$

    (b) Both insertion and bubble takes $\theta(n^2)$ in the worst case

    (c) Both insertion and bubble takes $O(nk)$ for all inputs.

    (d) None of the above

24. For an integer array $A = (a_1, a_2, \ldots, a_n)$, the inversion is a pair (a_i, a_j) such that $i < j$ and $a_i > a_j$. Suppose there are $\frac{n^2}{2}$ inversion pairs in A. Then,

    (a) Insertion sort takes at least $\frac{n^2}{2}$ swaps while sorting.

    (b) Insertion sort takes at most $\frac{n^2}{2}$ comparisons while sorting.

    (c) The function of insertion sort is independent of the number of inversion pairs

    (d) This is a best case input as the array is in almost sorted order.

25. Suppose we wish to use binary search module to identify the right position for $a[i]$ as $a[1..(i-1)]$ is already sorted. Then, what is the time complexity of insertion sort.

    (a) $T(n) = T(n-1) + \log n$

    (b) $T(n) = T(n-1) + \log n + O(1)$

    (c) $T(n) = T(n-1) + n \log n$

    (d) $T(n) = T(n-1) + \log n + O(n)$

26. Consider this Hybrid Sorting; the first $\frac{n}{2}$ elements are sorted using bubble sort and the second half using insertion sort. Suppose the given array is in almost sorted order, then what is the overall time complexity of hybrid sorting

    (a) $O(n)$ for this input, for other inputs, $O(n^2)$

    (b) $O(n^2)$ for this input and all inputs.

    (c) $O(n)$ for all inputs.

    (d) None of the above.

27. The worst case complexity of stable in-place sorting is

    (a) $\Theta(n \log n)$ time, $O(n)$ space

    (b) $\Theta(n^2)$ time, $O(n)$ space

    (c) $\Theta(n \log n)$ time, $\Theta(n \log n)$ space

    (d) $\Theta(n^2)$ time and space

28. The objective is to find k smallest elements in an integer array using quick sort partition routine as a black box (you cannot modify the partition routine, however, pre (post) processing of the input to the partition routine is allowed). Then,

    (a) $O(k)$ calls with overall time complexity $O(n^2)$

    (b) $O(n)$ calls with overall time complexity $O(n^2)$

    (c) $O(k)$ calls with overall time complexity $O(nk)$

    (d) $O(k)$ calls with overall time complexity $O(n \log n)$

29. Suppose you are given a subroutine which will output the median of n elements in $O(n)$ time. Then, what is the worst case time complexity of quick sort,

    (a) $T(n) = 2T(\frac{n}{2}) + O(n) + O(n)$

    (b) $T(n) = T(k) + T(n-k-1) + O(n) + O(n)$, for some $k \geq 0$.

    (c) $T(n) = T(n-1) + O(n)$

    (d) None of the above

30. Which of the following is(are) true

    (a) Quick sort is an unstable sorting algorithm for some inputs

    (b) If the array is either in sorted order or reverse sorted order, then Quick sort take $\Theta(n^2)$.

    (c) The element of the small window (elements smaller than pivot) is never compared with element of the large window.

    (d) All of the above