- Question# 2, 3, 5, 8 are new that were not present in previous question set with file name ***Some Question on Data Structure Autumn2020.pdf*** shared on 19-Aug-2020 on WhatsApp Group DAA Autumn2020.
- Question# 14 in the shared file was misprinted and became identical to question# 13. This has been corrected. Questions# 13 and 14 of the previously shared file have become questions# 17 and 18 in this file.

Some Questions on Data Structure (26-Aug-2020):

1. What is data?
2. What is a Data Type?
3. What is difference between primitive (built-in) and derived data types? Give examples.
4. What is data structure?
5. What operations are performed on a data structure (derived data types)? Some of them are traversing, search, etc.
6. What is linear and non-linear data structure?
7. Why array, stack, queue are called linear data structure while tree, graphs are called non-linear data structure?
8. What are static and dynamic data structures?
9. What is an array data structure?
10. What is two/three/multi-dimensional array? How are they implemented in C language?
11. What is row major order and column major order of data storage for two dimensional arrays?
12. What is a dynamic array?

13. What is a linked list?
14. What is a doubly linked list—its advantage and disadvantage over normal linked list?
15. What is a circular linked list—its advantage and disadvantage over normal linked list?
16. Compare advantages and disadvantages of arrays with those of linked lists.

17. How will you store polynomial like $A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ in memory using arrays? Write pseudo codes to (i) add/subtract (ii) multiply two such polynomials $A(x)$ and $B(x)$ and store the result in a new polynomial $C(x)$.
18. How will you store polynomial like $A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ in memory using linked lists? Write pseudo codes to (i) add/subtract (ii) multiply two such polynomials $A(x)$ and $B(x)$ and store the result in a new polynomial $C(x)$.
19. Compare advantages and disadvantages of array versus linked list representation of polynomials.

20. What is a stack data structure? What operations are performed on stacks?
21. How will you implement a stack using an array? Write pseudo codes for PUSH and POP operations on a stack using arrays.
22. How will you implement a stack using a linked list? Write pseudo codes for PUSH and POP operations on a stack using linked lists.
23. Compare array implementation versus linked list implementation of stacks.
24. What are the applications of stacks in computer programming? Mention any three.

25. What is a queue data structure? What operations are performed on queues?
26. How will you implement a queue using an array? Write pseudo codes for INSERT and DELETE operations on queue using arrays.

27. What is a circular queue? Why this queue is preferred over straight linear queue in array based implementation?

28. Write pseudo codes for INSERT and DELETE operations on circular queue using arrays.

29. How will you implement a queue using a linked list? Write pseudo codes for INSERT and DELETE operations on queue using linked lists.

30. Compare array implementation versus linked list implementation of queues.

31. What are the applications of queues in computer programming? Mention any three.

32. Is it possible to simulate a stack using two queues? If yes, then how? Write the pseudo code for PUSH and POP operations of a stack S simulated using the INSERT and DELETE operations of two queues Q1 and Q2.

33. Is it possible to simulate a queue using two stacks? If yes, then how? Write the pseudo code for INSERT and DELETE operations of a queue Q simulated using the PUSH and POP operations of two stacks S1 and S2.

34. What is tree data structure? Define it formally.

35. How can a tree be stored in the main memory of a computer?

36. What operations can be performed on a tree? Write their pseudo codes?

37. Mention some applications of a tree data structure in building computer software.

38. What is a binary tree? Define it formally.

39. How can a binary tree be stored in the main memory of a computer?

40. What operations can be performed on a tree? Write their pseudo codes?

41. Write pseudo codes for different types of traversals of a binary tree.

42. Mention some applications of a tree data structure in computer programming.

43. How many distinct binary trees are possible with n-nodes? Derive a formula for this.

44. Derive an expression for relationship between height of tree and maximum as well as minimum number of leaf nodes.

45. **Using method of induction, prove that**

Lemma 5.1:

(i) The maximum number of nodes on level i of a binary tree is 2^{i-1}, $i \geq 1$ and

(ii) The maximum number of nodes in a binary tree of depth k is $2^k - 1$, $k \geq 1$.

46. **Lemma 5.2:** For any nonempty binary tree, T, if n_0 is the number of terminal nodes and n_2 the number of nodes of degree 2,

then $n_0 = n_2 + 1$.

47. **Lemma 5.3:** If a complete binary tree with n nodes (i.e., depth= $[\log_2 n] + 1$) is represented with sequential node numbers, that is, numbering starts from the root, numbering at next level starts only after numbering all the nodes at the current level from left to right in sequence, then for any node with index i, $1 \leq i \leq n$, we have:

(i) PARENT(i) is at $[i/2]$ if $i \geq 1$. When $i = 1$, i is the root and has no parent.

(ii) LCHILD(i) is at $2i$ if $2i \leq n$. If $2i >$ n, then i has no left child.

(iii) RCHILD(i) is at $2i + 1$ if $2i + 1 \leq n$. If $2i + 1 > n$, then i has no right child.

48. What is array representation of a binary tree? What are its disadvantages?

49. What are decision trees?

50. Consider the well-known *eight coins* problem. Given coins *a,b,c,d,e,f,g,h,* we are told that one is a counterfeit and has a different weight than the others. We want to determine which coin it is, making use of an equal arm balance. We want to do so using a minimum number of comparisons and at the same time determine whether the false coin is heavier or lighter than the rest. Draw a decision tree to solve this problem and write pseudo code for the solution using this.

51. How disjoint sets may be represented as trees? Write efficient algorithms UNION (to merge two sets) and FIND (to determine the root of the tree to which a set-element belongs) for these sets.

52. **Lemma 5.5:** Let T be a tree with n nodes created as a result of algorithm UNION. Show that no node in T has level greater $\lfloor \log_2 n + 1 \rfloor$.

Note: (i) The function Floor(x) expressed as $\lfloor x \rfloor$ is defined as the highest integer less than or equal to x.
For example, $\lfloor 2.4 \rfloor = 2$, $\lfloor 2.8 \rfloor = 2$, $\lfloor 2.99 \rfloor = 2$, $\lfloor 2 \rfloor = 2$.. It simply ignores the fractional component of a number.
(ii) The function Ceiling(x) expressed as $\lceil x \rceil$ is defined as the lowest integer greater than or equal to x. For example, $\lceil 2.001 \rceil = 3$, $\lceil 2.8 \rceil = 3$, $\lceil 2.99 \rceil = 3$, $\lceil 2 \rceil = 2$. It forces to take the full item when a part of it is essentially required. If we need to hire minimum 3.4 people, then we would have to hire 4 people.

53. What are game trees? Discuss one such tree with necessary figure and pseudo codes.

54. What are binary search trees (BSTs)?

55. Write pseudo codes to perform INSERT, DELETE and SEARCH operations on BSTs.

56. Why do we need to balance a BST?

57. Why is balanced BST also called an AVL tree?

58. Illustrate with examples and clear figures the four types of rotations LL, RR, LR and RL required to balance a BST.

59. Write pseudo codes to perform INSERT, DELETE and SEARCH operations on AVL trees (balanced BSTs).

60. What is a ***heap*** data structure?

61. What are the applications of **heap**?

62. Write pseudo codes to perform various operations on a heap and find their time complexities.

63. What do you mean by asymptotic time complexity of an operation or an algorithm? Why do we use such technique to specify the time taken by an algorithm?

64. Define big O, Ω, and θ notations to specify the time complexities of algorithms with necessary figures and formula. .

65. Solve the recurrence relation $T(n) = 2T(n/2) + c$, where n is a variable and c is a constant and $T(1) = 1$.

66. Show that $T(n) = 2T(n/2) + n = \theta(n \log_2 n)$, where n is a variable and $T(1) = 1$.

67. Solve the recurrence relation $T(n) = 2T(n/4) + n$, where n is a variable and $T(1) = 1$

68. Show that $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$ is $O(n \lg n)$, where $\lg n$ stands for $\log_2 n$.

69. Show the derivation of the recursion tree for $T(n) = 3T(n/4) + cn^2$.

70. Show the derivation of the recursion tree for $T(n) = 2T(n/2) + n$.

71. Show the derivation of the recursion tree for $T(n) = 3T(n/4) + cn^2$.

72. Solve using recursion tree for $T(n) = T(n/3) + T(2n/3) + O(n)$, where $T(1) = 1$.

73. Solve $T(n) = 3T(\lfloor n/2 \rfloor) + n$.

74. Draw the recursion tree for $T(n) = 4T(\lfloor n/2 \rfloor) + cn$, where c is a constant, and provide a tight asymptotic bound on its solution.

75. Use a recursion tree to give an asymptotically tight solution to the recurrence $T(n) = T(n - a) + T(a) + cn$, where $a \geq 1$ and $c > 0$ are constants.

76. Use a recursion tree to give an asymptotically tight solution to the recurrence $T(n) = T(\alpha n) + T((1 - \alpha)n) + cn$, where α is a constant in the range $0 < \alpha < 1$ and $c > 0$ is also a constant.

77. Solve $T(n) = 2T(n/2) + n \lg n$, where $\lg n$ stands for $\log_2 n$.

78. Argue that the solution to the recurrence $T(n) = T(n/3) + T(2n/3) + cn$, where c is a constant, is $\Omega(n \lg n)$ by appealing to a recursion tree.

79. Solve the recurrences

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

b. $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$

c. $T(n) = 9T(n/3) + n$

d. $T(n) = T(\sqrt{n}) + 1$

e. $T(n) = T(n/4) + \sqrt{n}$

f. $T(n) = 3T(n/2) + n \lg n$.

g. $T(n) = 4T(n/2) + n^2 \lg n$

h. $T(n) = 2T(n/2) + n^3$.

i. $T(n) = T(9n/10) + n$.

j. $T(n) = 16T(n/4) + n^2$.

k. $T(n) = 7T(n/3) + n^2$.

l. $T(n) = 7T(n/2) + n^2$

m. $T(n) = 5T(n/5) + n/ \lg n$.

n. $T(n) = 4T(n/2) + n^2\sqrt{n}$

o. $T(n) = 3T(n/3 + 5) + n/2$.

p. $T(n) = 2T(n/2) + n/ \lg n$.

q. $T(n) = T(n/2) + T(n/4) + T(n/8) + n$.

r. $T(n) = T(n - 1) + 1/n$.

s. h. $T(n) = 4T(n/2) + \sqrt{n}$

t. $T(n) = T(n - 1) + \lg n$.

u. $T(n) = T(n - 2) + 2 \lg n$

v. $T(n) = \sqrt{n}T(\sqrt{n}) + n$.

w. $T(n) = 2T(\lfloor \sqrt{n} \rfloor) + \lg n$.

More questions may be given after few weeks.