Data Structure Interview Questions

1. What is a data structure?

The data structure is the way data is organized (stored) and manipulated for retrieval and access. It also defines the way different sets of data relate to one another, establishing relationships and forming algorithms.

2. What is a linear data structure? Name a few examples.

A data structure is linear if all its elements or data items are arranged in a sequence or a linear order. The elements are stored in a non-hierarchical way so that each item has successors and predecessors except the first and last element in the list.

Examples of linear data structures are Arrays, Stack, Strings, Queue, and Linked List.

3. What are some applications of data structures?

Numerical analysis, operating system, AI, compiler design, database management, graphics, statistical analysis, and simulation.

4. What is the difference between file structure and storage structure?

The difference lies in the memory area accessed. Storage structure refers to the data structure in the memory of the computer system, whereas file structure represents the storage structure in the auxiliary memory.

5. What is a linked list data structure?

It's a linear data structure or a sequence of data objects where elements are not stored in adjacent memory locations. The elements are linked using pointers to form a chain. Each element is a separate object, called a node. Each node has two items: a data field and a reference to the next node. The entry point in a linked list is called the head. Where the list is empty, the head is a null reference and the last node has a reference to null.

A linked list is a dynamic data structure, where the number of nodes is not fixed, and the list has the ability to grow and shrink on demand.

It is applied in cases where:

- We deal with an unknown number of objects or don't know how many items are in the list
- We need constant-time insertions/deletions from the list, as in real-time computing where time predictability is critical
- Random access to any elements is not needed
- The algorithm requires a data structure where objects need to be stored irrespective of their physical address in memory
- We need to insert items in the middle of the list as in a priority queue

Some implementations are stacks and queues, graphs, directory of names, dynamic memory allocation, and performing arithmetic operations on long integers.

6. Are linked lists considered linear or non-linear data structures?

Linked lists are considered both linear and non-linear data structures depending upon the application they are used for. When used for access strategies, it is considered as a linear data-structure. When used for data storage, it is considered a non-linear data structure.

7. What are the advantages of a linked list over an array? In which scenarios do we use Linked List and when Array?

Advantages of a linked list over an array are:

1. Insertion and Deletion

Insertion and deletion of nodes is an easier process, as we only update the address present in the next pointer of a node. It's expensive to do the same in an array as the room has to be created for the new elements and existing elements must be shifted.

2. Dynamic Data Structure

As a linked list is a dynamic data structure, there is no need to give an initial size as it can grow and shrink at runtime by allocating and deallocating memory. However, the size is limited in an array as the number of elements is statically stored in the main memory.

3. No wastage of memory

As the size of a linked list can increase or decrease depending on the demands of the program, and memory is allocated only when required, there is no memory wasted. In the case of an array, there is memory wastage. For instance, if we declare an array of size 10 and store only five elements in it, then the space for five elements is wasted.

4. Implementation

Data structures like stack and queues are more easily implemented using a linked list than an array.

Some scenarios where we use linked list over array are:

- When we know the upper limit on the number of elements in advance
- When there are a large number of add or remove operations
- When there are no large number of random access to elements
- When we want to insert items in the middle of the list, such as when implementing a priority queue

Some scenarios in which we use array over the linked list are:

- When we need to index or randomly access elements
- When we know the number of elements in the array beforehand, so we can allocate the correct amount of memory
- When we need speed when iterating through all the elements in the sequence
- When memory is a concern; filled arrays use less memory than linked lists, as each element in the array is the data but each linked list node requires the data as well as one or more pointers to the other elements in the linked list

In summary, we consider the requirements of space, time, and ease of implementation to decide whether to use a linked list or array.

8. What is a doubly-linked list? Give some examples.

It is a complex type (double-ended LL) of a linked list in which a node has two links, one that connects to the next node in the sequence and another that connects to the previous node. This allows traversal across the data elements in both directions.

Examples include:

- A music playlist with next and previous navigation buttons
- The browser cache with BACK-FORWARD visited pages
- The undo and redo functionality on a browser, where you can reverse the node to get to the previous page

9. How do you reference all of the elements in a one-dimension array?

All of the elements in a one-dimension array can be referenced using an indexed loop as the array subscript so that the counter runs from 0 to the array size minus one.

10. What are dynamic data structures? Name a few.

They are collections of data in memory that expand and contract to grow or shrink in size as a program runs. This enables the programmer to control exactly how much memory is to be utilized.

Examples are the dynamic array, linked list, stack, queue, and heap.

11. What is an algorithm?

An algorithm is a step by step method of solving a problem or manipulating data. It defines a set of instructions to be executed in a certain order to get the desired output.

12. Why do we need to do an algorithm analysis?

A problem can be solved in more than one way using several solution algorithms. Algorithm analysis provides an estimation of the required resources of an algorithm to solve a specific computational problem. The amount of time and space resources required to execute is also determined.

The time complexity of an algorithm quantifies the amount of time taken for an algorithm to run as a function of the length of the input. The space complexity quantifies the amount of space or memory taken by an algorithm, to run as a function of the length of the input.

13. What is a stack?

A stack is an abstract data type that specifies a linear data structure, as in a real physical stack or piles where you can only take the top item off the stack in order to remove things. Thus, insertion

(push) and deletion (pop) of items take place only at one end called top of the stack, with a particular order: LIFO (Last In First Out) or FILO (First In Last Out).

14. Where are stacks used?

- Expression, evaluation, or conversion of evaluating prefix, postfix, and infix expressions
- Syntax parsing
- String reversal
- Parenthesis checking
- Backtracking

15. What is a queue data structure?

A queue is an abstract data type that specifies a linear data structure or an ordered list, using the First In First Out (FIFO) operation to access elements. Insert operations can be performed only at one end called REAR and delete operations can be performed only at the other end called FRONT.

16. List some applications of queue data structure.

To prioritize jobs as in the following scenarios:

- As waiting lists for a single shared resource in a printer, CPU, call center systems, or image uploads; where the first one entered is the first to be processed
- In the asynchronous transfer of data; or example pipes, file IO, and sockets
- As buffers in applications like MP3 media players and CD players
- To maintain the playlist in media players (to add or remove the songs)

17. What is a Dequeue?

It is a double-ended queue, or a data structure, where the elements can be inserted or deleted at both ends (FRONT and REAR).

18. What operations can be performed on queues?

- enqueue() adds an element to the end of the queue
- dequeue() removes an element from the front of the queue
- init() is used for initializing the queue
- isEmpty tests for whether or not the queue is empty
- The front is used to get the value of the first data item but does not remove it
- The rear is used to get the last item from a queue

19. What are the advantages of the heap over a stack?

Generally, both heap and stack are part of memory and used in Java for different needs:

- Heap is more flexible than the stack because memory space can be dynamically allocated and de-allocated as needed
- Heap memory is used to store objects in Java, whereas stack memory is used to store local variables and function call
- Objects created in the heap are visible to all threads, whereas variables stored in stacks are only visible to the owner as private memory
- When using recursion, the size of heap memory is more whereas it quickly fill-ups stack memory

20. Where can stack data structure be used?

- Expression evaluation
- Backtracking
- Memory management
- Function calling and return

21. What is the difference between a PUSH and a POP?

The acronyms stand for Pushing and Popping operations performed on a stack. These are ways data is stored and retrieved.

- PUSH is used to add an item to a stack, while POP is used to remove an item.

- PUSH takes two arguments, the name of the stack to add the data to and the value of the entry to be added. POP only needs the name of the stack.
- When the stack is filled and another PUSH command is issued, you get a stack overflow error, which means that the stack can no longer accommodate the last PUSH. In POP, a stack underflow error occurs when you're trying to POP an already empty stack.

22. Which sorting algorithm is considered the fastest? Why?

A single sorting algorithm can't be considered best, as each algorithm is designed for a particular data structure and data set. However, the QuickSort algorithm is generally considered the fastest because it has the best performance for most inputs.

Its advantages over other sorting algorithms include the following:

- Cache-efficient: It linearly scans and linearly partitions the input. This means we can make the most of every cache load.
- Can skip some swaps: As QuickSort is slightly sensitive to input that is in the right order, it can skip some swaps.
- Efficient even in worst-case input sets, as the order is generally random.
- Easy adaption to already- or mostly-sorted inputs.
- When speed takes priority over stability.

23. What is the merge sort? How does it work?

Merge sort is a divide-and-conquer algorithm for sorting the data. It works by merging and sorting adjacent data to create bigger sorted lists, which are then merged recursively to form even bigger sorted lists until you have one single sorted list.

24. How does the Selection sort work?

Selection sort works by repeatedly picking the smallest number in ascending order from the list and placing it at the beginning. This process is repeated moving toward the end of the list or sorted subarray.

Scan all items and find the smallest. Switch over the position as the first item. Repeat the selection sort on the remaining N-1 items. We always iterate forward (i from 0 to N-1) and swap with the smallest element (always i).

Time complexity: best case O(n2); worst O(n2)

Space complexity: worst O(1)

25. Define the graph data structure?

It is a type of non-linear data structure that consists of vertices or nodes connected by edges or arcs to enable storage or retrieval of data. Edges may be directed or undirected.

26. What are the applications of graph data structure?

- Transport grids where stations are represented as vertices and routes as the edges of the graph
- Utility graphs of power or water, where vertices are connection points and edge the wires or pipes connecting them
- Social network graphs to determine the flow of information and hotspots (edges and vertices)
- Neural networks where vertices represent neurons and edge the synapses between them

27. What are the advantages of binary search over a linear search?

In a sorted list:

- A binary search is more efficient than a linear search because we perform fewer comparisons. With linear search, we can only eliminate one element per comparison each time we fail to find the value we are looking for, but with the binary search, we eliminate half the set with each comparison.
- Binary search runs in O(log n) time compared to linear search's O(n) time. This means that the more of the elements present in the search array, the faster is binary search compared to a linear search.

28. What is an AVL tree?

An AVL (Adelson, Velskii, and Landi) tree is a height balancing binary search tree in which the difference of heights of the left and right subtrees of any node is less than or equal to one. This controls the height of the binary search tree by not letting it get skewed. This is used when working with a large data set, with continual pruning through insertion and deletion of data.

29. Differentiate NULL and VOID

- Null is a value, whereas Void is a data type identifier
- Null indicates an empty value for a variable, whereas void indicates pointers that have no initial size
- Null means it never existed; Void means it existed but is not in effect

30. Do dynamic memory allocations help in managing data? How?

Dynamic memory allocation stores simple structured data types at runtime. It has the ability to combine separately allocated structured blocks to form composite structures that expand and contract as needed, thus helping manage data of data blocks of arbitrary size, in arbitrary order.

31. Name the ways to determine whether a linked list has a loop.

- Using hashing
- Using the visited nodes method (with or without modifying the basic linked list data structure)
- Floyd’s cycle-finding algorithm

32. List some applications of multilinked structures?

- Sparse matrix
- Index generation

33. Explain the jagged array.

It is an array whose elements themselves are arrays and may be of different dimensions and sizes.

34. Explain the max heap data structure.

It is a type of heap data structure where the value of the root node is greater than or equal to either of its child nodes.

35. How do you find the height of a node in a tree?

The height of the node equals the number of edges in the longest path to the leaf from the node, where the depth of a leaf node is 0.

1. What is data structure?
Ans: The logical and mathematical model of a particular organization of data is called data structure.
There are two types of data structure
i) Linear
ii) Nonlinear

2. What are the goals of Data Structure?
Ans: It must rich enough in structure to reflect the actual relationship of data in real world.
The structure should be simple enough for efficient processing of data.

3. What does abstract Data Type Mean?
Ans: Data type is a collection of values and a set of operations on these values. Abstract data type refer to the mathematical concept that define the data type.
It is a useful tool for specifying the logical properties of a data type.
ADT consists of two parts
1) Values definition
2) Operation definition
Example:-The value definition for the ADT RATIONAL states that RATIONAL value consists of two integers, second doesn't equal to zero.
The operator definition for ADT RATIONAL includes the operation of creation (make rational) addition, multiplication and test for equality.

4. What is the difference between a Stack and an Array?
Ans:
i) Stack is a ordered collection of items
ii) Stack is a dynamic object whose size is constantly changing as items are pushed and popped .
iii) Stack may contain different data types
iv) Stack is declared as a structure containing an array to hold the element of the stack, and an integer to indicate the current stack top within the array.
ARRAY
i) Array is an ordered collection of items
ii) Array is a static object i.e. no of item is fixed and is assigned by the declaration of the array
iii) It contains same data types.
iv) Array can be home of a stack i.e. array can be declared large enough for maximum size of the stack.

5. What do you mean by recursive definition?
Ans: The definition which defines an object in terms of simpler cases of itself is called recursive definition.

6. What is sequential search?
Ans: In sequential search each item in the array is compared with the item being searched until a match occurs. It is applicable to a table organized either as an array or as a linked list.

7. What actions are performed when a function is called?
Ans: When a function is called
i) arguments are passed
ii) local variables are allocated and initialized
ii) transferring control to the function

8. What actions are performed when a function returns?
Ans:
i) Return address is retrieved
ii) Function's data area is freed
iii) Branch is taken to the return address

9. What is a linked list?
Ans: A linked list is a linear collection of data elements, called nodes, where the linear order is given by pointers. Each node has two parts first part contain the information of the element second part contains the address of the next node in the list.

10. What are the advantages of linked list over array (static data structure)?
Ans:
The disadvantages of array are
i) unlike linked list it is expensive to insert and delete elements in the array
ii) One can't double or triple the size of array as it occupies block of memory space.

In linked list
i) each element in list contains a field, called a link or pointer which contains the address of the next element
ii) Successive element's need not occupy adjacent space in memory.

21. What are the disadvantages of linear list?
Ans:
i) We cannot reach any of the nodes that precede node (p)
ii) If a list is traversed, the external pointer to the list must be persevered in order to reference the list again

22. Define circular list?
Ans: In linear list the next field of the last node contain a null pointer, when a next field in the last node contain a pointer back to the first node it is called circular list.
Advantages – From any point in the list it is possible to reach at any other point

23. What are the disadvantages of circular list?
Ans:
i) We can't traverse the list backward
ii) If a pointer to a node is given we cannot delete the node

24. Define double linked list?
Ans: It is a collection of data elements called nodes, where each node is divided into three parts
i) An info field that contains the information stored in the node
ii) Left field that contain pointer to node on left side
iii) Right field that contain pointer to node on right side

25. Is it necessary to sort a file before searching a particular item ?
Ans:
If less work is involved in searching a element than to sort and then extract, then we don't go for sort
If frequent use of the file is required for the purpose of retrieving specific element, it is more efficient to sort the file.
Thus it depends on situation.

26. What are the issues that hamper the efficiency in sorting a file?
Ans: The issues are
i) Length of time required by the programmer in coding a particular sorting program
ii) Amount of machine time necessary for running the particular program
iii)The amount of space necessary for the particular program .

27. Calculate the efficiency of sequential search?
Ans: The number of comparisons depends on where the record with the argument key appears in the table
If it appears at first position then one comparison
If it appears at last position then n comparisons
Average=(n+1)/2 comparisons
Unsuccessful search n comparisons
Number of comparisons in any case is O (n).

28. Is any implicit arguments are passed to a function when it is called?
Ans: Yes there is a set of implicit arguments that contain information necessary for the function to execute and return correctly. One of them is return address which is stored within the function's

data area, at the time of returning to calling program the address is retrieved and the function branches to that location.

29. Parenthesis is never required in Postfix or Prefix expressions, why?
Ans: Parenthesis is not required because the order of the operators in the postfix /prefix expressions determines the actual order of operations in evaluating the expression

30. List out the areas in which data structures are applied extensively?
Ans:
Compiler Design,
Operating System,
Database Management System,
Statistical analysis package,
Numerical Analysis,
Graphics,
Artificial Intelligence,
Simulation

11. Can we apply binary search algorithm to a sorted linked list, why?
Ans: No we cannot apply binary search algorithm to a sorted linked list, since there is no way of indexing the middle element in the list. This is the drawback in using linked list as a data structure.

11. Can we apply binary search algorithm to a sorted linked list, why?
Ans: No we cannot apply binary search algorithm to a sorted linked list, since there is no way of indexing the middle element in the list. This is the drawback in using linked list as a data structure.

12. What do you mean by free pool?
Ans: Pool is a list consisting of unused memory cells which has its own pointer.

13. What do you mean by garbage collection?
Ans: It is a technique in which the operating system periodically collects all the deleted space onto the free storage list.
It takes place when there is minimum amount of space left in storage list or when CPU is ideal.

The alternate method to this is to immediately reinsert the space into free storage list which is time consuming.

14. What do you mean by overflow and underflow?
Ans: When new data is to be inserted into the data structure but there is no available space i.e. free storage list is empty this situation is called overflow.
When we want to delete data from a data structure that is empty this situation is called underflow.

15. What are the disadvantages array implementations of linked list?
Ans:
i) The no of nodes needed can't be predicted when the program is written.
ii) The no of nodes declared must remain allocated throughout its execution

16. What is a queue?
Ans: A queue is an ordered collection of items from which items may be deleted at one end (front end) and items inserted at the other end (rear end).
It obeys FIFO rule there is no limit to the number of elements a queue contains.

17. What is a priority queue?
Ans: The priority queue is a data structure in which the intrinsic ordering of the elements (numeric or alphabetic)
Determines the result of its basic operation. It is of two types
i) Ascending priority queue- Here smallest item can be removed (insertion is arbitrary)
ii) Descending priority queue- Here largest item can be removed (insertion is arbitrary)

18. What are the disadvantages of sequential storage?
Ans:
i) Fixed amount of storage remains allocated to the data structure even if it contains less element.
ii) No more than fixed amount of storage is allocated causing overflow

19. What are the disadvantages of representing a stack or queue by a linked list?
Ans:
i) A node in a linked list (info and next field) occupies more storage than a corresponding element in an array.
ii) Additional time spent in managing the available list.

20. What is dangling pointer and how to avoid it?
Ans: After a call to free(p) makes a subsequent reference to *p illegal, i.e. though the storage to p is freed but the value of p(address) remain unchanged .so the object at that address may be used as the value of *p (i.e. there is no way to detect the illegality).Here p is called dangling pointer.

To avoid this it is better to set p to NULL after executing free(p).The null pointer value doesn't reference a storage location it is a pointer that doesn't point to anything.

12. What do you mean by free pool?
Ans: Pool is a list consisting of unused memory cells which has its own pointer.

13. What do you mean by garbage collection?
Ans: It is a technique in which the operating system periodically collects all the deleted space onto the free storage list.
It takes place when there is minimum amount of space left in storage list or when CPU is ideal.
The alternate method to this is to immediately reinsert the space into free storage list which is time consuming.

14. What do you mean by overflow and underflow?
Ans: When new data is to be inserted into the data structure but there is no available space i.e. free storage list is empty this situation is called overflow.
When we want to delete data from a data structure that is empty this situation is called underflow.

15. What are the disadvantages array implementations of linked list?
Ans:
i) The no of nodes needed can't be predicted when the program is written.
ii) The no of nodes declared must remain allocated throughout its execution

16. What is a queue?
Ans: A queue is an ordered collection of items from which items may be deleted at one end (front end) and items inserted at the other end (rear end).
It obeys FIFO rule there is no limit to the number of elements a queue contains.

17. What is a priority queue?
Ans: The priority queue is a data structure in which the intrinsic ordering of the elements (numeric or alphabetic)
Determines the result of its basic operation. It is of two types
i) Ascending priority queue- Here smallest item can be removed (insertion is arbitrary)
ii) Descending priority queue- Here largest item can be removed (insertion is arbitrary)

18. What are the disadvantages of sequential storage?
Ans:

i) Fixed amount of storage remains allocated to the data structure even if it contains less element.
ii) No more than fixed amount of storage is allocated causing overflow

19. What are the disadvantages of representing a stack or queue by a linked list?
Ans:
i) A node in a linked list (info and next field) occupies more storage than a corresponding element in an array.
ii) Additional time spent in managing the available list.

20. What is dangling pointer and how to avoid it?
Ans: After a call to free(p) makes a subsequent reference to *p illegal, i.e. though the storage to p is freed but the value of p(address) remain unchanged .so the object at that address may be used as the value of *p (i.e. there is no way to detect the illegality).Here p is called dangling pointer.
To avoid this it is better to set p to NULL after executing free(p).The null pointer value doesn't reference a storage location it is a pointer that doesn't point to anything.

31. What are the major data structures used in the following areas : network data model & Hierarchical data model.
Ans:
RDBMS – Array (i.e. Array of structures)
Network data model – Graph
Hierarchical data model – Trees

32. If you are using C language to implement the heterogeneous linked list, what pointer type will you use?
Ans: The heterogeneous linked list contains different data types in its nodes and we need a link, pointer to connect them. It is not possible to use ordinary pointers for this. So we go for void pointer. Void pointer is capable of storing pointer to any type as it is a generic pointer type.

33. Minimum number of queues needed to implement the priority queue?
Ans: Two. One queue is used for actual storing of data and another for storing priorities.

34. What is the data structures used to perform recursion?
Ans: Stack. Because of its LIFO (Last In First Out) property it remembers its 'caller' so knows whom to return when the function has to return. Recursion makes use of system stack for storing the return addresses of the function calls.
Every recursive function has its equivalent iterative (non-recursive) function. Even when such equivalent iterative procedures are written, explicit stack is to be used.

35. What are the notations used in Evaluation of Arithmetic Expressions using prefix and postfix forms?
Ans: Polish and Reverse Polish notations.

36. Convert the expression ((A + B) * C – (D – E) ^ (F + G)) to equivalent Prefix and Postfix notations.
Ans: Prefix Notation:
^ – * +ABC – DE + FG
Postfix Notation:
AB + C * DE – – FG + ^

37. Sorting is not possible by using which of the following methods?
(a) Insertion
(b) Selection
(c) Exchange
(d) Deletion

Ans: (d) Deletion.
Using insertion we can perform insertion sort, using selection we can perform selection sort, using exchange we can perform the bubble sort (and other similar sorting methods). But no sorting method can be done just using deletion.

38. List out few of the Application of tree data-structure?
Ans:
The manipulation of Arithmetic expression,
Symbol Table construction,
Syntax analysis.

39. List out few of the applications that make use of Multilinked Structures?
Ans: Sparse matrix, Index generation.

40. in tree construction which is the suitable efficient data structure?
(A) Array (b) Linked list (c) Stack (d) Queue (e) none
Ans: (b) Linked list

41. What is the type of the algorithm used in solving the 8 Queens problem?
Ans: Backtracking

42. In an AVL tree, at what condition the balancing is to be done?
Ans: If the 'pivotal value' (or the 'Height factor') is greater than 1 or less than –1.

43. There are 8, 15, 13, 14 nodes were there in 4 different trees. Which of them could have formed a full binary tree?
Ans: 15
In general:
There are 2n-1 nodes in a full binary tree.

By the method of elimination:
Full binary trees contain odd number of nodes. So there cannot be full binary trees with 8 or 14 nodes, so rejected. With 13 nodes you can form a complete binary tree but not a full binary tree. So the correct answer is 15.
Note: Full and Complete binary trees are different. All full binary trees are complete binary trees but not vice versa.

44. In RDBMS, what is the efficient data structure used in the internal storage representation?
Ans: B+ tree. Because in B+ tree, all the data is stored only in leaf nodes, that makes searching easier. This corresponds to the records that shall
be stored in leaf nodes.

45. One of the following tree structures, which is, efficient considering space and time complexities?
a) Incomplete Binary Tree.
b) Complete Binary Tree.
c) Full Binary Tree.
Ans:
b) Complete Binary Tree.
By the method of elimination:
Full binary tree loses its nature when operations of insertions and deletions are done. For incomplete binary trees,
extra property of complete binary tree is maintained even after operations like additions and deletions are done on it.

46. What is a spanning Tree?
Ans: A spanning tree is a tree associated with a network. All the nodes of the graph appear on the tree once. A minimum spanning tree is a spanning tree organized so that the total edge weight between nodes is minimized.

47. Does the minimum spanning tree of a graph give the shortest distance between any 2 specified nodes?
Ans: No.

Minimal spanning tree assures that the total weight of the tree is kept at its minimum. But it doesn't mean that the distance between any two nodes involved in the minimum-spanning tree is minimum.

48. Whether Linked List is linear or Non-linear data structure?
Ans: According to Storage Linked List is a Non-linear one.

1. **Question 1. What Is Data Structure?**
   Answer :
   A data structure is a way of organizing data that considers not only the items stored, but also their relationship to each other. Advance knowledge about the relationship between data items allows designing of efficient algorithms for the manipulation of data.
2. **Question 2. What Are The Goals Of Data Structure?**
   Answer :
   It must rich enough in structure to reflect the actual relationship of data in real world. The structure should be simple enough for efficient processing of data.

   RDBMS Interview Questions
3. **Question 3. What Does Abstract Data Type Mean?**
   Answer :
   Data type is a collection of values and a set of operations on these values. Abstract data type refer to the mathematical concept that define the data type.

   It is a useful tool for specifying the logical properties of a data type.

   ADT consists of two parts

   1) Values definition
   2) Operation definition
   Example:-The value definition for the ADT RATIONAL states that RATIONAL value consists of two integers, second doesn't equal to zero.

   The operator definition for ADT RATIONAL includes the operation of creation (make rational) addition, multiplication and test for equality.
4. **Question 4. What Is The Difference Between A Stack And An Array?**
   Answer :
   STACK:
   i) Stack is a ordered collection of items.
   ii) Stack is a dynamic object whose size is constantly changing as items are pushed and popped.
   iii) Stack may contain different data types.
   iv) Stack is declared as a structure containing an array to hold the element of the stack, and an integer to indicate the current stack top within the array.

ARRAY:

i) Array is an ordered collection of items.
ii) Array is a static object i.e. no of item is fixed and is assigned by the declaration of the array.
iii) It contains same data types.
iv) Array can be home of a stack i.e. array can be declared large enough for maximum size of the stack.

Adv Java Tutorial

5. **Question 5. What Do You Mean By Recursive Definition?**
   Answer :
   The definition which defines an object in terms of simpler cases of itself is called recursive definition.

   DBMS Interview Questions

6. **Question 6. What Is Sequential Search?**
   Answer :
   In sequential search each item in the array is compared with the item being searched until a match occurs. It is applicable to a table organized either as an array or as a linked list.
7. **Question 7. What Actions Are Performed When A Function Is Called?**
   Answer :
   When a function is called

   i) arguments are passed.
   ii) local variables are allocated and initialized.
   ii) transferring control to the function.

   Core Java Tutorial | Adv Java Interview Questions

8. **Question 8. What Actions Are Performed When A Function Returns?**
   Answer :
   i) Return address is retrieved.
   ii) Function's data area is freed.
   iii) Branch is taken to the return address.
9. **Question 9. What Is A Linked List?**
   Answer :
   A linked list is a linear collection of data elements, called nodes, where the linear order is given by pointers. Each node has two parts first part contain the information of the element second part contains the address of the next node in the list.

   Core Java Interview Questions

10. **Question 10. What Are The Advantages Of Linked List Over Array (static Data Structure)?**
    Answer :

The disadvantages of array are:
i) unlike linked list it is expensive to insert and delete elements in the array.
ii) One can't double or triple the size of array as it occupies block of memory space.
In linked list
i) each element in list contains a field, called a link or pointer which contains the address of the next element.
ii) Successive element's need not occupy adjacent space in memory.

C Tutorial

11. **Question 11. We Apply Binary Search Algorithm To A Sorted Linked List, Why?**
Answer :
No we cannot apply binary search algorithm to a sorted linked list, since there is no way of indexing the middle element in the list. This is the drawback in using linked list as a data structure.

C Interview Questions

12. **Question 12. What Do You Mean By Free Pool?**
Answer :
Pool is a list consisting of unused memory cells which has its own pointer.

RDBMS Interview Questions

13. **Question 13. What Do You Mean By Garbage Collection?**
Answer :
It is a technique in which the operating system periodically collects all the deleted space onto the free storage list.

It takes place when there is minimum amount of space left in storage list or when CPU is ideal.

The alternate method to this is to immediately reinsert the space into free storage list which is time consuming.

14. **Question 14. What Do You Mean By Overflow And Underflow?**
Answer :
When new data is to be inserted into the data structure but there is no available space i.e. free storage list is empty this situation is called overflow.

When we want to delete data from a data structure that is empty this situation is called underflow.

15. **Question 15. What Are The Disadvantages Array Implementations Of Linked List?**
Answer :
i) The no of nodes needed can't be predicted when the program is written.
ii) The no of nodes declared must remain allocated throughout its execution.

16. **Question 16. What Is A Queue?**
    Answer :
    A queue is an ordered collection of items from which items may be deleted at one end (front end) and items inserted at the other end (rear end).

    It obeys FIFO rule there is no limit to the number of elements a queue contains.
17. **Question 17. What Is A Priority Queue?**
    Answer :
    The priority queue is a data structure in which the intrinsic ordering of the elements (numeric or alphabetic)

    Determines the result of its basic operation. It is of two types:

    i) Ascending priority queue- Here smallest item can be removed (insertion is arbitrary).
    ii) Descending priority queue- Here largest item can be removed (insertion is arbitrary).
18. **Question 18. What Are The Disadvantages Of Sequential Storage?**
    Answer :
    i) Fixed amount of storage remains allocated to the data structure even if it contains less element.
    ii) No more than fixed amount of storage is allocated causing overflow.
19. **Question 19. What Are The Disadvantages Of Representing A Stack Or Queue By A Linked List?**
    Answer :
    i) A node in a linked list (info and next field) occupies more storage than a corresponding element in an array.
    ii) Additional time spent in managing the available list.
20. **Question 20. What Is Dangling Pointer And How To Avoid It?**
    Answer :
    After a call to free(p) makes a subsequent reference to *p illegal, i.e. though the storage to p is freed but the value of p(address) remain unchanged .so the object at that address may be used as the value of *p (i.e. there is no way to detect the illegality).Here p is called dangling pointer.

    To avoid this it is better to set p to NULL after executing free(p).The null pointer value doesn’t reference a storage location it is a pointer that doesn’t point to anything.

21. **Question 21. What Are The Disadvantages Of Linear List?**
    Answer :
    i) We cannot reach any of the nodes that precede node (p).
    ii) If a list is traversed, the external pointer to the list must be persevered in order to reference the list again.
22. **Question 22. Define Circular List?**
    Answer :
    In linear list the next field of the last node contain a null pointer, when a next field in the last node contain a pointer back to the first node it is called circular list.

Advantages – From any point in the list it is possible to reach at any other point.

23. **Question 23. What Are The Disadvantages Of Circular List?**
    Answer :
    i) We can't traverse the list backward.
    ii) If a pointer to a node is given we cannot delete the node.
24. **Question 24. Define Double Linked List?**
    Answer :
    It is a collection of data elements called nodes,

    where each node is divided into three parts:
    - An info field that contains the information stored in the node.
    - Left field that contain pointer to node on left side.
    - Right field that contain pointer to node on right side.
25. **Question 25. Is It Necessary To Sort A File Before Searching A Particular Item ?**
    Answer :
    If less work is involved in searching a element than to sort and then extract, then we don't go for sort.

    If frequent use of the file is required for the purpose of retrieving specific element, it is more efficient to sort the file.

    Thus it depends on situation.
26. **Question 26. What Are The Issues That Hamper The Efficiency In Sorting A File?**
    Answer :
    The issues are:
    - Length of time required by the programmer in coding a particular sorting program.
    - Amount of machine time necessary for running the particular program.
    - The amount of space necessary for the particular program .

27. **Question 27. Calculate The Efficiency Of Sequential Search?**
    Answer :
    The number of comparisons depends on where the record with the argument key appears in the table.

```
If it appears at first position then one comparison

If it appears at last position then n comparisons

Average=(n+1)/2 comparisons

Unsuccessful search n comparisons

Number of comparisons in any case is O (n).
```

28. **Question 28. Is Any Implicit Arguments Are Passed To A Function When It Is Called?**
    Answer :

Yes there is a set of implicit arguments that contain information necessary for the function to execute and return correctly. One of them is return address which is stored within the function's data area, at the time of returning to calling program the address is retrieved and the function branches to that location.

29. **Question 29. Parenthesis Is Never Required In Postfix Or Prefix Expressions, Why?**
    Answer :
    Parenthesis is not required because the order of the operators in the postfix /prefix expressions determines the actual order of operations in evaluating the expression.
30. **Question 30. List Out The Areas In Which Data Structures Are Applied Extensively?**
    Answer :
    - Compiler Design,
    - Operating System,
    - Database Management System,
    - Statistical analysis package,
    - Numerical Analysis,
    - Graphics,
    - Artificial Intelligence,
    - Simulation.
31. **Question 31. What Are The Major Data Structures Used In The Following Areas : Network Data Model & Hierarchical Data Model?**
    Answer :
    RDBMS – Array (i.e. Array of structures)
    Network data model – Graph
    Hierarchical data model – Trees
32. **Question 32. If You Are Using C Language To Implement The Heterogeneous Linked List, What Pointer Type Will You Use?**
    Answer :
    The heterogeneous linked list contains different data types in its nodes and we need a link, pointer to connect them. It is not possible to use ordinary pointers for this. So we go for void pointer. Void pointer is capable of storing pointer to any type as it is a generic pointer type.

33. **Question 33. Minimum Number Of Queues Needed To Implement The Priority Queue?**
    Answer :
    Two. One queue is used for actual storing of data and another for storing priorities.
34. **Question 34. What Is The Data Structures Used To Perform Recursion?**
    Answer :
    Stack. Because of its LIFO (Last In First Out) property it remembers its 'caller' so knows whom to return when the function has to return. Recursion makes use of system stack for storing the return addresses of the function calls.

    Every recursive function has its equivalent iterative (non-recursive) function. Even when such equivalent iterative procedures are written, explicit stack is to be used.
35. **Question 35. What Are The Notations Used In Evaluation Of Arithmetic Expressions Using Prefix And Postfix Forms?**
    Answer :
    Polish and Reverse Polish notations.

36. **Question 36. Convert The Expression ((a + B) * C - (d - E) ^ (f + G)) To Equivalent Prefix And Postfix Notations?**
    Answer :
    Prefix Notation:
    ^ – * +ABC – DE + FG

    postfix Notation:
    AB + C * DE – – FG + ^
37. **Question 37. List Out Few Of The Application Of Tree Data-structure?**
    Answer :
    The manipulation of Arithmetic expression, Symbol Table construction & Syntax analysis.
38. **Question 38. List Out Few Of The Applications That Make Use Of Multilinked Structures?**
    Answer :
    Sparse matrix, Index generation.
39. **Question 39. What Is The Type Of The Algorithm Used In Solving The 8 Queens Problem?**
    Answer :
    Backtracking.
40. **Question 40. In An Avl Tree, At What Condition The Balancing Is To Be Done?**
    Answer :
    If the 'pivotal value' (or the 'Height factor') is greater than 1 or less than –1.
41. **Question 41. There Are 8, 15, 13, 14 Nodes Were There In 4 Different Trees. Which Of Them Could Have Formed A Full Binary Tree?**
    Answer :
    In general: There are 2n-1 nodes in a full binary tree. By the method of elimination:

    Full binary trees contain odd number of nodes. So there cannot be full binary trees with 8 or 14 nodes, so rejected. With 13 nodes you can form a complete binary tree but not a full binary tree. So the correct answer is 15.
42. **Question 42. In Rdbms, What Is The Efficient Data Structure Used In The Internal Storage Representation?**
    Answer :
    B+ tree. Because in B+ tree, all the data is stored only in leaf nodes, that makes searching easier. This corresponds to the records that shall be stored in leaf nodes.
43. **Question 43. What Is A Spanning Tree?**
    Answer :
    A spanning tree is a tree associated with a network. All the nodes of the graph appear on the tree once. A minimum spanning tree is a spanning tree organized so that the total edge weight between nodes is minimized.
44. **Question 44. Does The Minimal Spanning Tree Of A Graph Give The Shortest Distance Between Any 2 Specified Nodes?**
    Answer :
    No! Minimal spanning tree assures that the total weight of the tree is kept at its minimum. But it doesn't mean that the distance between any two nodes involved in the minimal-spanning tree is minimum.

45. **Question 45. Difference Between Calloc And Malloc ?**
    Answer :
    malloc: allocate n bytes.
    calloc: allocate m times n bytes initialized to 0.
46. **Question 46. What Are The Major Data Structures Used In The Following Areas : Rdbms, Network Data Model & Hierarchical Data Model?**
    Answer :
    - RDBMS Array (i.e. Array of structures)
    - Network data model Graph
    - Hierarchical data model Trees.
47. **Question 47. Which File Contains The Definition Of Member Functions?**
    Answer :
    Definitions of member functions for the Linked List class are contained in the LinkedList.cpp file.

48. **Question 48. How Is Any Data Structure Application Is Classified Among Files?**
    Answer :
    A linked list application can be organized into a header file, source file and main application file. The first file is the header file that contains the definition of the NODE structure and the LinkedList class definition. The second file is a source code file containing the implementation of member functions of the LinkedList class. The last file is the application file that contains code that creates and uses the LinkedList class.
49. **Question 49. What Member Function Places A New Node At The End Of The Linked List?**
    Answer :
    The appendNode() member function places a new node at the end of the linked list. The appendNode() requires an integer representing the current data of the node.
50. **Question 50. What Is Linked List ?**
    Answer :
    Linked List is one of the fundamental data structures. It consists of a sequence of ? nodes, each containing arbitrary data fields and one or two (”links”) pointing to the next and/or previous nodes. A linked list is a self-referential datatype because it contains a pointer or link to another data of the same type. Linked lists permit insertion and removal of nodes at any point in the list in constant time, but do not allow random access.
51. **Question 51. What Does Each Entry In The Link List Called?**
    Answer :
    Each entry in a linked list is called a node. Think of a node as an entry that has three sub entries. One sub entry contains the data, which may be one attribute or many attributes. Another points to the previous node, and the last points to the next node. When you enter a new item on a linked list, you allocate the new node and then set the pointers to previous and next nodes.
52. **Question 52. How Is The Front Of The Queue Calculated ?**
    Answer :
    The front of the queue is calculated by front = (front+1) % size.

53. **Question 53. Why Is The Isempty() Member Method Called?**
    Answer :
    The isEmpty() member method is called within the dequeue process to determine if there is an item in the queue to be removed i.e. isEmpty() is called to decide whether the queue has at least one element. This method is called by the dequeue() method before returning the front element.
54. **Question 54. Which Process Places Data At The Back Of The Queue?**
    Answer :
    Enqueue is the process that places data at the back of the queue.
55. **Question 55. What Is The Relationship Between A Queue And Its Underlying Array?**
    Answer :
    Data stored in a queue is actually stored in an array. Two indexes, front and end will be used to identify the start and end of the queue.

    When an element is removed front will be incremented by 1. In case it reaches past the last index available it will be reset to 0. Then it will be checked with end. If it is greater than end queue is empty.

    When an element is added end will be incremented by 1. In case it reaches past the last index available it will be reset to 0. After incrementing it will be checked with front. If they are equal queue is full.
56. **Question 56. What Is A Queue ?**
    Answer :
    A Queue is a sequential organization of data. A queue is a first in first out type of data structure. An element is inserted at the last position and an element is always taken out from the first position.
57. **Question 57. What Does Isempty() Member Method Determines?**
    Answer :
    isEmpty() checks if the stack has at least one element. This method is called by Pop() before retrieving and returning the top element.
58. **Question 58. What Method Removes The Value From The Top Of A Stack?**
    Answer :
    The pop() member method removes the value from the top of a stack, which is then returned by the pop() member method to the statement that calls the pop() member method.
59. **Question 59. What Method Is Used To Place A Value Onto The Top Of A Stack?**
    Answer :
    push() method, Push is the direction that data is being added to the stack. push() member method places a value onto the top of a stack.
60. **Question 60. Run Time Memory Allocation Is Known As ?**
    Answer :
    Allocating memory at runtime is called a dynamically allocating memory. In this, you dynamically allocate memory by using the new operator when declaring the array.

    for example : int grades[] = new int[10];
61. **Question 61. How Do You Assign An Address To An Element Of A Pointer Array ?**
    Answer :

We can assign a memory address to an element of a pointer array by using the address operator, which is the ampersand (&), in an assignment statement such as ptemployee[0] = &projects[2];

62. **Question 62. Why Do We Use A Multidimensional Array?**
    Answer :
    A multidimensional array can be useful to organize subgroups of data within an array. In addition to organizing data stored in elements of an array, a multidimensional array can store memory addresses of data in a pointer array and an array of pointers.

    Multidimensional arrays are used to store information in a matrix form.
    e.g; a railway timetable, schedule cannot be stored as a single dimensional array. One can use a 3-D array for storing height, width and length of each room on each floor of a building.

63. **Question 63. What Is Significance Of " * " ?**
    Answer :
    The symbol "*" tells the computer that you are declaring a pointer. Actually it depends on context.

    In a statement like int *ptr; the '*' tells that you are declaring a pointer.
    In a statement like int i = *ptr; it tells that you want to assign value pointed to by ptr to variable i.
    The symbol "*" is also called as Indirection Operator/ Dereferencing Operator.
64. **Question 64. Is Pointer A Variable?**
    Answer :
    Yes, a pointer is a variable and can be used as an element of a structure and as an attribute of a class in some programming languages such as C++, but not Java. However, the contents of a pointer is a memory address of another location of memory, which is usually the memory address of another variable, element of a structure, or attribute of a class.
65. **Question 65. How Many Parts Are There In A Declaration Statement?**
    Answer :
    There are two main parts, variable identifier and data type and the third type is optional which is type qualifier like signed/unsigned.
66. **Question 66. How Memory Is Reserved Using A Declaration Statement ?**
    Answer :
    Memory is reserved using data type in the variable declaration. A programming language implementation has predefined sizes for its data types.

    For example:
    in C# the declaration int i; will reserve 32 bits for variable i.
    A pointer declaration reserves memory for the address or the pointer variable, but not for the data that it will point to. The memory for the data pointed by a pointer has to be allocated at runtime.
    The memory reserved by the compiler for simple variables and for storing pointer address is

allocated on the stack, while the memory allocated for pointer referenced data at runtime is allocated on the heap.

67. **Question 67. What Is Impact Of Signed Numbers On The Memory?**
    Answer :
    Sign of the number is the first bit of the storage allocated for that number. So you get one bit less for storing the number. For example if you are storing an 8-bit number, without sign, the range is 0-255. If you decide to store sign you get 7 bits for the number plus one bit for the sign. So the range is -128 to +127.
68. **Question 68. What Is Precision?**
    Answer :
    Precision refers the accuracy of the decimal portion of a value. Precision is the number of digits allowed after the decimal point.
69. **Question 69. What Is The Difference Between Null And Void Pointer?**
    Answer :
    NULL can be value for pointer type variables.
    VOID is a type identifier which has not size.
    NULL and void are not same. Example: void* ptr = NULL;
70. **Question 70. What Is The Difference Between Array And Stack?**
    Answer :
    STACK follows LIFO. Thus the item that is first entered would be the last removed.

    In array the items can be entered or removed in any order. Basically each member access is done using index. No strict order is to be followed here to remove a particular element.

    Array may be multidiamensional or onediamensional but stack should be onediamensional. but both are linear data structure.
71. **Question 71. Tell How To Check Whether A Linked List Is Circular ?**
    Answer :
    Create two pointers, each set to the start of the list. Update each as follows:

```
while (pointer1)

{

pointer1 = pointer1->next;

pointer2 = pointer2->next;

if(pointer2)pointer2=pointer2->next;

if (pointer1 == pointer2)

{

print (”circularn”);

}

}
```

72. **Question 72. Whether Linked List Is Linear Or Non-linear Data Structure?**
    Answer :
    - According to Access strategies Linked list is a linear one.

- According to Storage Linked List is a Non-linear one.

73. **Question 73. If You Are Using C Language To Implement The Heterogeneous Linked List, What Pointer Type Will You Use?**
    Answer :
    The heterogeneous linked list contains different data types in its nodes and we need a link, pointer to connect them. It is not possible to use ordinary pointers for this. So we go for void pointer. Void pointer is capable of storing pointer to any type as it is a generic pointer type.
74. **Question 74. What Is A Node Class?**
    Answer :
    A node class is a class that, relies on the base class for services and implementation, provides a wider interface to users than its base class, relies primarily on virtual functions in its public interface depends on all its direct and indirect base class can be understood only in the context of the base class can be used as base for further derivation can be used to create objects. A node class is a class that has added new services or functionality beyond the services inherited from its base class.
75. **Question 75. When Can You Tell That A Memory Leak Will Occur?**
    Answer :
    A memory leak occurs when a program loses the ability to free a block of dynamically allocated memory.

76. **Question 76. How Many Different Trees Are Possible With 10 Nodes ?**
    Answer :
    1014 - For example, consider a tree with 3 nodes(n=3), it will have the maximum combination of 5 different (ie, 23 - 3 =? 5) trees.
77. **Question 77. How Can I Search For Data In A Linked List?**
    Answer :
    Unfortunately, the only way to search a linked list is with a linear search, because the only way a linked list's members can be accessed is sequentially. Sometimes it is quicker to take the data from a linked list and store it in a different data structure so that searches can be more efficient.
78. **Question 78. Define Data Structures?**
    Answer :
    Data Structures is defined as the way of organizing all data items that consider not only the elements stored but also stores the relationship between the elements.
79. **Question 79. Define Primary Data Structures?**
    Answer :
    Primary data structures are the basic data structures that directly operate upon the machine instructions. All the basic constants (integers, floating-point numbers, character constants, string constants) and pointers are considered as primary data structures.
80. **Question 80. Define Static Data Structures?**
    Answer :
    A data structure formed when the number of data items are known in advance is referred as static data structure or fixed size data structure.
81. **Question 81. List Some Of The Static Data Structures In C?**

Answer :
Some of the static data structures in C are arrays, pointers, structures etc.

82. **Question 82. Define Dynamic Data Structures?**
Answer :
A data structure formed when the number of data items are not known in advance is known as dynamic data structure or variable size data structure.

83. **Question 83. List Some Of The Dynamic Data Structures In C?**
Answer :
Some of the dynamic data structures in C are linked lists, stacks, queues, trees etc.

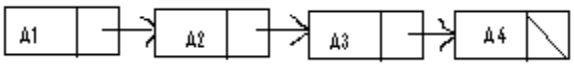
84. **Question 84. Define Linear Data Structures?**
Answer :
Linear data structures are data structures having a linear relationship between its adjacent elements.

Eg; Linked lists.

85. **Question 85. Define Non-linear Data Structures?**
Answer :
Non-linear data structures are data structures that don't have a linear relationship between its adjacent elements but have a hierarchical relationship between the elements.

Eg; Trees and Graphs.

86. **Question 86. Define Linked Lists?**
Answer :
Linked list consists of a series of structures, which are not necessarily adjacent in memory. Each structure contains the element and a pointer to a structure containing its successor. We call this the Next Pointer. The last cell's Next pointer points to NULL.



87. **Question 87. State The Different Types Of Linked Lists?**
Answer :
The different types of linked list include singly linked list, doubly linked list and circular linked list.

88. **Question 88. List The Basic Operations Carried Out In A Linked List?**
Answer :
The basic operations carried out in a linked list include:

- Creation of a list.
- Insertion of a node.
- Deletion of a node.
- Modification of a node.
- Traversal of the list.

89. **Question 89. List Out The Advantages Of Using A Linked List?**
Answer :
- It is not necessary to specify the number of elements in a linked list during its declaration.
- Linked list can grow and shrink in size depending upon the insertion and deletion that occurs in the list.

- Insertions and deletions at any place in a list can be handled easily and efficiently.
- A linked list does not waste any memory space.

90. **Question 90. List Out The Disadvantages Of Using A Linked List?**
Answer :
    - Searching a particular element in a list is difficult and time consuming.
    - A linked list will use more storage space than an array to store the same number of elements.
91. **Question 91. List Out The Applications Of A Linked List?**
Answer :
Some of the important applications of linked lists are manipulation of polynomials, sparse matrices, stacks and queues.
92. **Question 92. State The Difference Between Arrays And Linked Lists?**
Answer :
93. **Question 93. Define A Stack?**
Answer :
Stack is an ordered collection of elements in which insertions and deletions are restricted to one end. The end from which elements are added and/or removed is referred to as top of the stack. Stacks are also referred as piles, push-down lists and last-in-first-out (LIFO) lists.
94. **Question 94. List Out The Basic Operations That Can Be Performed On A Stack ?**
Answer :
The basic operations that can be performed on a stack are
    - Push operation.
    - Pop operation.
    - Peek operation.
    - Empty check.
    - Fully occupied check.
95. **Question 95. State The Different Ways Of Representing Expressions?**
Answer :
The different ways of representing expressions are
    - Infix Notation.
    - Prefix Notation.
    - Postfix Notation.
96. **Question 96. State The Advantages Of Using Infix Notations?**
Answer :
    - It is the mathematical way of representing the expression.
    - It is easier to see visually which operation is done from first to last.
97. **Question 97. State The Advantages Of Using Postfix Notations?**
Answer :
    - Need not worry about the rules of precedence.
    - Need not worry about the rules for right to left associativity.
    - Need not need parenthesis to override the above rules.
98. **Question 98. State The Rules To Be Followed During Infix To Postfix Conversions?**
Answer :
    - Fully parenthesize the expression starting from left to right. During parenthesizing, the operators having higher precedence are first parenthesized.
    - Move the operators one by one to their right, such that each operator replaces their corresponding right parenthesis.

- The part of the expression, which has been converted into postfix is to be treated as single operand.
- Once the expression is converted into postfix form, remove all parenthesis.

99. **Question 99. State The Rules To Be Followed During Infix To Prefix Conversions?**

Answer :

- Fully parenthesize the expression starting from left to right. During parenthesizing, the operators having higher precedence are first parenthesized.
- Move the operators one by one to their left, such that each operator replaces their corresponding left parenthesis.
- The part of the expression, which has been converted into prefix is to be treated as single operand.
- Once the expression is converted into prefix form, remove all parenthesis.

100. **Question 100. State The Difference Between Stacks And Linked Lists?**

Answer :

The difference between stacks and linked lists is that insertions and deletions may occur anywhere in a linked list, but only at the top of the stack.

101. **Question 101. Mention The Advantages Of Representing Stacks Using Linked Lists Than Arrays?**

Answer :

- It is not necessary to specify the number of elements to be stored in a stack during its declaration, since memory is allocated dynamically at run time when an element is added to the stack.
- Insertions and deletions can be handled easily and efficiently.
- Linked list representation of stacks can grow and shrink in size without wasting memory space, depending upon the insertion and deletion that occurs in the list.
- Multiple stacks can be represented efficiently using a chain for each stack.

102. **Question 102. Define A Queue?**

Answer :

Queue is an ordered collection of elements in which insertions are restricted to one end called the rear end and deletions are restricted to other end called the front end. Queues are also referred as First-In-First-Out (FIFO) Lists.

103. **Question 103. Define A Priority Queue?**

Answer :

Priority queue is a collection of elements, each containing a key referred as the priority for that element. Elements can be inserted in any order (i.e., of alternating priority), but are arranged in order of their priority value in the queue. The elements are deleted from the queue in the order of their priority (i.e., the elements with the highest priority is deleted first). The elements with the same priority are given equal importance and processed accordingly.

104. **Question 104. State The Difference Between Queues And Linked Lists?**

Answer :

The difference between queues and linked lists is that insertions and deletions may occur anywhere in the linked list, but in queues insertions can be made only in the rear end and deletions can be made only in the front end.

105. **Question 105. Define A Deque?**

Answer :

Deque (Double-Ended Queue) is another form of a queue in which insertions and deletions are made at both the front and rear ends of the queue. There are two variations of a deque,

namely, input restricted deque and output restricted deque. The input restricted deque allows insertion at one end (it can be either front or rear) only. The output restricted deque allows deletion at one end (it can be either front or rear) only.

106. **Question 106. Why You Need A Data Structure?**

Answer :

A data structure helps you to understand the relationship of one data element with the other and organize it within the memory. Sometimes the organization might be simple and can be very clearly visioned.

Eg; List of names of months in a year –Linear Data Structure, List of historical places in the world- Non-Linear Data Structure. A data structure helps you to analyze the data, store it and organize it in a logical and mathematical manner.

107. **Question 107. What Do You Mean By Shortest Path?**

Answer :

A path having minimum weight between two vertices is known as shortest path, in which weight is always a positive number.

108. **Question 108. What Do You Mean By Articulation Point?**

Answer :

If a graph is not biconnected, the vertices whose removal would disconnect the graph are known as articulation points.

109. **Question 109. Define Biconnectivity?**

Answer :

A connected graph G is said to be biconnected, if it remains connected after removal of any one vertex and the edges that are incident upon that vertex. A connected graph is biconnected, if it has no articulation points.

110. **Question 110. What Do You Mean By Back Edge?**

Answer :

If w is the ancestor of v, then vw is called a back edge.

111. **Question 111. What Do You Mean By Tree Edge?**

Answer :

If w is undiscovered at the time vw is explored, then vw is called a tree edge and v becomes the parent of w.

112. **Question 112. Differentiate Bfs And Dfs?**

Answer :

113. **Question 113. What Do You Mean By Breadth First Search (bfs)?**

Answer :

BFS performs simultaneous explorations starting from a common point and spreading out independently.

114. **Question 114. List The Two Important Key Points Of Depth First Search?**

Answer :

i) If path exists from one node to another node, walk across the edge – exploring the edge.
ii) If path does not exist from one specific node to any other node, return to the previous node where we have been before – backtracking.

115. **Question 115. Define Graph Traversals?**

Answer :

Traversing a graph is an efficient way to visit each vertex and edge exactly once.

116. **Question 116. Name Two Algorithms Two Find Minimum Spanning Tree?**

Answer :

- Kruskal's algorithm.
- Prim's algorithm.

117. **Question 117. What Is A Minimum Spanning Tree?**

Answer :

A minimum spanning tree of an undirected graph G is a tree formed from graph edges that connects all the vertices of G at the lowest total cost.

118. **Question 118. What Are The Two Traversal Strategies Used In Traversing A Graph?**

Answer :

- Breadth first search
- Depth first search

119. **Question 119. When Is A Graph Said To Be Weakly Connected?**

Answer :

When a directed graph is not strongly connected but the underlying graph is connected, then the graph is said to be weakly connected.

120. **Question 120. What Is Meant By Strongly Connected In A Graph?**

Answer :

An undirected graph is connected, if there is a path from every vertex to every other vertex. A directed graph with this property is called strongly connected.

121. **Question 121. What Is An Acyclic Graph?**

Answer :

A simple diagram which does not have any cycles is called an acyclic graph.

122. **Question 122. What Is A Cycle Or A Circuit?**

Answer :

A path which originates and ends in the same node is called a cycle or circuit.

123. **Question 123. What Is A Simple Path?**

Answer :

A path in a diagram in which the edges are distinct is called a simple path. It is also called as edge simple.

124. **Question 124. Define Path In A Graph?**

Answer :

The path in a graph is the route taken to reach terminal node from a starting node.

125. **Question 125. Define Indegree Of A Graph?**

Answer :

In a directed graph, for any node v, the number of edges which have v as their terminal node is called the indegree of the node v.

126. **Question 126. Define Outdegree Of A Graph?**

Answer :

In a directed graph, for any node v, the number of edges which have v as their initial node is called the out degree of the node v.

127. **Question 127. What Is A Weighted Graph?**

Answer :

A graph in which weights are assigned to every edge is called a weighted graph.

128. **Question 128. What Is A Simple Graph?**

Answer :

A simple graph is a graph, which has not more than one edge between a pair of nodes than such a graph is called a simple graph.

129. **Question 129. What Is A Loop?**

Answer :

An edge of a graph which connects to itself is called a loop or sling.

130. **Question 130. What Is A Undirected Graph?**

Answer :

A graph in which every edge is undirected is called a directed graph.

131. **Question 131. What Is A Directed Graph?**

Answer :

A graph in which every edge is directed is called a directed graph.

132. **Question 132. Define Adjacent Nodes?**

Answer :

Any two nodes which are connected by an edge in a graph are called adjacent nodes. For example, if an edge x ε E is associated with a pair of nodes (u,v) where u, v ε V, then we say that the edge x connects the nodes u and v.

133. **Question 133. Define Graph?**

Answer :

A graph G consist of a nonempty set V which is a set of nodes of the graph, a set E which is the set of edges of the graph, and a mapping from the set for edge E to a set of pairs of elements of V. It can also be represented as G=(V, E).

134. **Question 134. What Is The Need For Path Compression?**

Answer :

Path compression is performed during a Find operation. Suppose if we want to perform Find(X), then the effect of path compression is that every node on the path from X to the root has its parent changed to the root.

135. **Question 135. What Do You Mean By Union-by-weight?**

Answer :

Keep track of the weight ie; size of each tree and always append the smaller tree to the larger one when performing UNION.

136. **Question 136. List The Abstract Operations In The Set?**

Answer :

Let S and T be sets and e be an element.

- SINGLETON(e) returns {e}.
- UNION(S,T) returns S Ụ T.
- INTERSECTION(S,T) returns S ∩ T.
- FIND returns the name of the set containing a given element.

2. **Question 137. Define A Set?**
   Answer :
   A set S is an unordered collection of elements from a universe. An element cannot appear more than once in S. The cardinality of S is the number of elements in S. An empty set is a set whose cardinality is zero. A singleton set is a set whose cardinality is one.
3. **Question 138. What Do You Mean By Disjoint Set Adt?**
   Answer :
   A collection of non-empty disjoint sets S=S1,S2,….,Sk i.e;each Si is a non-empty set that has no element in common with any other Sj. In mathematical notation this is: Si∩Sj=Φ. Each set is identified by a unique element called its representative.
4. **Question 139. List The Applications Of Set Adt?**
   Answer :
   - Maintaining a set of connected components of a graph.
   - Maintain list of duplicate copies of web pages.
   - Constructing a minimum spanning tree for a graph.
   -
5. **Question 140. Define An Equivalence Relation?**
   Answer :
   An equivalence relation is a relation R that satisfies three properties:
   - (Reflexive) aRa, for all a ε S.
   - (Symmetric) aRb if and only if bRa.
   - (Transitive) aRb and bRc implies that aRc.
6. **Question 141. Define A Relation?**
   Answer :
   A relation R is defined on a set S if for every pair of elements (a,b), a,b ε S, aRb is either true or false. If aRb is true, then we say that a is related to b.
7. **Question 142. Mention One Advantage And Disadvantage Of Using Quadratic Probing?**
   Answer :
   Advantage: The problem of primary clustering is eliminated.
   Disadvantage: There is no guarantee of finding an unoccupied cell once the table is nearly half full.
8. **Question 143. List The Limitations Of Linear Probing?**
   Answer :
   - Time taken for finding the next available cell is large.
   - In linear probing, we come across a problem known as clustering.
9. **Question 144. What Is The Need For Extendible Hashing?**
   Answer :
   If either open addressing hashing or separate chaining hashing is used, the major problem is that collisions could cause several blocks to be examined during a Find, even for a well-distributed hash table. Extendible hashing allows a find to be performed in two disk accesses. Insertions also require few disk accesses.
10. **Question 145. What Do You Mean By Rehashing?**
    Answer :
    If the table gets too full, the running time for the operations will start taking too long and inserts might fail for open addressing with quadratic resolution. A solution to this is to build another table that is about twice as big with the associated new hash function and scan down

the entire original hash table, computing the new hash value for each element and inserting it in the new table. This entire operation is called rehashing.

11. **Question 146. What Do You Mean By Double Hashing?**
    Answer :
    Double hashing is an open addressing collision resolution strategy in which F(i)=i.hash2(X). This formula says that we apply a second hash function to X and probe at a distance hash2(X), 2hash2(X),….,and so on. A function such as hash2(X)=R-(XmodR), with R a prime smaller than Tablesize.
12. **Question 147. What Do You Mean By Secondary Clustering?**
    Answer :
    Although quadratic probing eliminates primary clustering, elements that hash to the same position will probe the same alternative cells. This is known as secondary clustering.
13. **Question 148. What Do You Mean By Quadratic Probing?**
    Answer :
    Quadratic probing is an open addressing collision resolution strategy in which F(i)=i2. There is no guarantee of finding an empty cell once the table gets half full if the table size is not prime. This is because at most half of the table can be used as alternative locations to resolve collisions.
14. **Question 149. What Do You Mean By Primary Clustering?**
    Answer :
    In linear probing collision resolution strategy, even if the table is relatively empty, blocks of occupied cells start forming. This effect is known as primary clustering means that any key hashes into the cluster will require several attempts to resolve the collision and then it will add to the cluster.
15. **Question 150. What Do You Mean By Linear Probing?**
    Answer :
    Linear probing is an open addressing collision resolution strategy in which F is a linear function of i, F(i)=i. This amounts to trying sequentially in search of an empty cell. If the table is big enough, a free cell can always be found, but the time to do so can get quite large.
16. **Question 151. What Do You Mean By Probing?**
    Answer :
    Probing is the process of getting next available hash table array cell.
17. **Question 152. What Are The Types Of Collision Resolution Strategies In Open Addressing?**
    Answer :
    - Linear probing.
    - Quadratic probing.
    - Double hashing.
18. **Question 153. What Do You Mean By Open Addressing?**
    Answer :
    Open addressing is a collision resolving strategy in which, if collision occurs alternative cells are tried until an empty cell is found. The cells h0(x), h1(x), h2(x),…. are tried in succession, where hi(x)=(Hash(x)+F(i))mod Tablesize with F(0)=0. The function F is the collision resolution strategy.
19. **Question 154. Write The Disadvantages Of Separate Chaining?**

Answer :

- The elements are evenly distributed. Some elements may have more elements and some may not have anything.
- It requires pointers. This leads to slow the algorithm down a bit because of the time required to allocate new cells, and also essentially requires the implementation of a second data structure.

20. **Question 155. Write The Advantage Of Separate Chaining?**
    Answer :
    More number of elements can be inserted as it uses linked lists.
21. **Question 156. What Do You Mean By Separate Chaining?**
    Answer :
    Separate chaining is a collision resolution technique to keep the list of all elements that hash to the same value. This is called separate chaining because each hash table element is a separate chain (linked list). Each linked list contains all the elements whose keys hash to the same index.
22. **Question 157. What Are The Collision Resolution Methods?**
    Answer :
    - Separate chaining or External hashing.
    - Open addressing or Closed hashing.
23. **Question 158. What Do You Mean By Collision In Hashing?**
    Answer :
    When an element is inserted, it hashes to the same value as an already inserted element, and then it produces collision.
24. **Question 159. Write The Importance Of Hashing?**
    Answer :
    - Maps key with the corresponding value using hash function.
    - Hash tables support the efficient addition of new entries and the time spent on searching for the required data is independent of the number of items stored.
25. **Question 160. What Do You Mean By Hash Function?**
    Answer :
    A hash function is a key to address transformation which acts upon a given key to compute the relative position of the key in an array. The choice of hash function should be simple and it must distribute the data evenly. A simple hash function is hash_key=key mod tablesize.
26. **Question 161. What Do You Mean By Hash Table?**
    Answer :
    The hash table data structure is merely an array of some fixed size, containing the keys. A key is a string with an associated value. Each key is mapped into some number in the range 0 to tablesize-1 and placed in the appropriate cell.
27. **Question 162. Define Hashing?**
    Answer :
    Hashing is the transformation of string of characters into a usually shorter fixed length value or key that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the short hashed key than to find it using the original value.

28. **Question 163. What Do You Mean By The Term "percolate Down"?**
    Answer :
    When the minimum element is removed, a hole is created at the root. Since the heap now becomes one smaller, it follows that the last element X in the heap must move somewhere in the heap. If X can be placed in the hole, then we are done.. This is unlikely, so we slide the smaller of the hole's children into the hole, thus pushing the hole down one level. We repeat this step until X can be placed in the hole. Thus, our action is to place X in its correct spot along a path from the root containing minimum children. This general strategy is known as percolate down.
29. **Question 164. What Do You Mean By The Term "percolate Up"?**
    Answer :
    To insert an element, we have to create a hole in the next available heap location. Inserting an element in the hole would sometimes violate the heap order property, so we have to slide down the parent into the hole. This strategy is continued until the correct location for the new element is found. This general strategy is known as a percolate up; the new element is percolated up the heap until the correct location is found.

30. **Question 165. What Are The Applications Of Priority Queues?**
    Answer :
    - The selection problem.
    - Event simulation.
31. **Question 166. What Do You Mean By Heap Order Property?**
    Answer :
    In a heap, for every node X, the key in the parent of X is smaller than (or equal to) the key in X, with the exception of the root (which has no parent).
32. **Question 167. What Do You Mean By Structure Property In A Heap?**
    Answer :
    A heap is a binary tree that is completely filled with the possible exception at the bottom level, which is filled from left to right. Such a tree is known as a complete binary tree.
33. **Question 168. What Are The Properties Of Binary Heap?**
    Answer :
    - Structure Property.
    - Heap Order Property.
34. **Question 169. What Is The Need For Priority Queue?**
    Answer :
    In a multiuser environment, the operating system scheduler must decide which of the several processes to run only for a fixed period of time. One algorithm uses queue. Jobs are initially placed at the end of the queue. The scheduler will repeatedly take the first job on the queue, run it until either it finishes or its time limit is up and place it at the end of the queue if it does not finish. This strategy is not appropriate, because very short jobs will soon to take a long time because of the wait involved in the run.

    Generally, it is important that short jobs finish as fast as possible, so these jobs should have precedence over jobs that have already been running. Further more, some jobs that are not short are still very important and should have precedence. This particular application seems to

require a special kind of queue, known as priority queue. Priority queue is also called as Heap or Binary Heap.

35. **Question 170. What Are The Applications Of B-tree?**
    Answer :
    - Database implementation.
    - Indexing on non primary key fields.
36. **Question 171. What Do You Mean By 2-3-4 Tree?**
    Answer :
    A B-tree of order 4 is called 2-3-4 tree. A B-tree of order 4 is a tree that is not binary with the following structural properties:
    - The root is either a leaf or has between 2 and 4 children.
    - All non-leaf nodes (except the root) have between 2 and 4 children.
    - All leaves are at the same depth.
37. **Question 172. What Do You Mean By 2-3 Tree?**
    Answer :
    A B-tree of order 3 is called 2-3 tree. A B-tree of order 3 is a tree that is not binary with the following structural properties:
    - The root is either a leaf or has between 2 and 3 children.
    - All non-leaf nodes (except the root) have between 2 and 3 children.
    - All leaves are at the same depth.
38. **Question 173. Define B-tree Of Order M?**
    Answer :
    A B-tree of order M is a tree that is not binary with the following structural properties:
    - The root is either a leaf or has between 2 and M children.
    - All non-leaf nodes (except the root) have between [M/2] and M children.
    - All leaves are at the same depth.
39. **Question 174. What Is The Minimum Number Of Nodes In An Avl Tree Of Height H?**
    Answer :
    The minimum number of nodes S(h), in an AVL tree of height h is given by S(h)=S(h-1)+S(h-2)+1. For h=0, S(h)=1.
40. **Question 175. Define Heap?**
    Answer :
    A heap is defined to be a complete binary tree with the property that the value of each node is atleast as small as the value of its child nodes, if they exist. The root node of the heap has the smallest value in the tree.
41. **Question 176. List The Types Of Rotations Available In Splay Tree?**
    Answer :
    Let us assume that the splay is performed at vertex v, whose parent and grandparent are p and g respectively. Then, the three rotations are named as:

    Zig: If p is the root and v is the left child of p, then left-left rotation at p would suffice. This case always terminates the splay as v reaches the root after this rotation.
    Zig-Zig: If p is not the root, p is the left child and v is also a left child, then a left-left rotation at g followed by a left-left rotation at p, brings v as an ancestor of g as well as p.
    Zig-Zag: If p is not the root, p is the left child and v is a right child, perform a left-right rotation at g and bring v as an ancestor of p as well as g.

42. **Question 177. What Is The Idea Behind Splaying?**
    Answer :
    Splaying reduces the total accessing time if the most frequently accessed node is moved towards the root. It does not require to maintain any information regarding the height or balance factor and hence saves space and simplifies the code to some extent.
43. **Question 178. Define Splay Tree?**
    Answer :
    A splay tree is a binary search tree in which restructuring is done using a scheme called splay. The splay is a heuristic method which moves a given vertex v to the root of the splay tree using a sequence of rotations.
44. **Question 179. What Do You Mean By Balance Factor Of A Node In Avl Tree?**
    Answer :
    The height of left subtree minus height of right subtree is called balance factor of a node in AVL tree.The balance factor may be either 0 or +1 or -1.The height of an empty tree is -1.
45. **Question 180. What Are The Categories Of Avl Rotations?**
    Answer :
    Let A be the nearest ancestor of the newly inserted nod which has the balancing factor ±2. Then the rotations can be classified into the following four categories:

    Left-Left: The newly inserted node is in the left subtree of the left child of A.
    Right-Right: The newly inserted node is in the right subtree of the right child of A.
    Left-Right: The newly inserted node is in the right subtree of the left child of A.
    Right-Left: The newly inserted node is in the left subtree of the right child of A.
46. **Question 181. What Do You Mean By Balanced Trees?**
    Answer :
    Balanced trees have the structure of binary trees and obey binary search tree properties. Apart from these properties, they have some special constraints, which differ from one data structure to another. However, these constraints are aimed only at reducing the height of the tree, because this factor determines the time complexity.

    Eg: AVL trees, Splay trees.
47. **Question 182. Define Avl Tree?**
    Answer :
    An empty tree is height balanced. If T is a non-empty binary tree with TL and TR as its left and right subtrees, then T is height balanced if
    - TL and TR are height balanced and
    - $|hL - hR| \leq 1$

    Where hL and hR are the heights of TL and TR respectively.
48. **Question 183. Define Left-in Threaded Tree?**
    Answer :
    Left-in threaded binary tree is defined as one in which each NULL pointers is altered to contain a thread to that node's inorder predecessor.
49. **Question 184. Define Right-in Threaded Tree?**
    Answer :
    Right-in threaded binary tree is defined as one in which threads replace NULL pointers in nodes with empty right sub-trees.

50. **Question 185. What Is An Expression Tree?**
    Answer :
    An expression tree is a tree which is build from infix or prefix or postfix expression. Generally, in such a tree, the leaves are operands and other nodes are operators.
51. **Question 186. What Is The Use Of Threaded Binary Tree?**
    Answer :
    In threaded binary tree, the NULL pointers are replaced by some addresses. The left pointer of the node points to its predecessor and the right pointer of the node points to its successor.
52. **Question 187. Why It Is Said That Searching A Node In A Binary Search Tree Is Efficient Than That Of A Simple Binary Tree?**
    Answer :
    In binary search tree, the nodes are arranged in such a way that the left node is having less data value than root node value and the right nodes are having larger value than that of root. Because of this while searching any node the value of the target node will be compared with the parent node and accordingly either left sub branch or right sub branch will be searched. So, one has to compare only particular branches. Thus searching becomes efficient.
53. **Question 188. Define Ancestor And Descendant ?**
    Answer :
    If there is a path from node n1 to n2, then n1 is the ancestor of n2 and n2 is the descendant of n1.
54. **Question 189. What Do You Mean By General Trees?**
    Answer :
    General tree is a tree with nodes having any number of children.
55. **Question 190. Define A Binary Search Tree?**
    Answer :
    A binary search tree is a special binary tree, which is either empty or it should satisfy the following characteristics:
    - Every node has a value and no two nodes should have the same value i.e) the values in the binary search tree are distinct.
    - The values in any left sub-tree is less than the value of its parent node.
    - The values in any right sub-tree is greater than the value of its parent node.
    - The left and right sub-trees of each node are again binary search trees.
56. **Question 191. State The Demerits Of Linked Representation Of Binary Trees?**
    Answer :
    - Given a node structure, it is difficult to determine its parent node.
    - Memory spaces are wasted for storing null pointers for the nodes, which have one or no sub-trees.
    - It requires dynamic memory allocation, which is not possible in some programming language.
57. **Question 192. State The Merit Of Linked Representation Of Binary Trees?**
    Answer :
    Insertions and deletions in a node involve no data movement except the rearrangement of pointers, hence less processing time.
58. **Question 193. State The Demerit Of Linear Representation Of Binary Trees?**
    Answer :

Insertions and deletions in a node take an excessive amount of processing time due to data movement up and down the array.

59. **Question 194. State The Merits Of Linear Representation Of Binary Trees?**
    Answer :
    - Storage method is easy and can be easily implemented in arrays.
    - When the location of a parent/child node is known, other one can be determined easily.
    - It requires static memory allocation so it is easily implemented in all programming language.
60. **Question 195. What Are The Tasks Performed During Postorder Traversal?**
    Answer :
    - Traverse the left sub-tree.
    - Traverse the right sub-tree.
    - Process the root node.
61. **Question 196. What Are The Tasks Performed During Inorder Traversal?**
    Answer :
    - Traverse the left sub-tree.
    - Process the root node.
    - Traverse the right sub-tree.
62. **Question 197. What Are The Tasks Performed During Preorder Traversal?**
    Answer :
    - Process the root node.
    - Traverse the left sub-tree.
    - Traverse the right sub-tree.
63. **Question 198. What Are The Tasks Performed While Traversing A Binary Tree?**
    Answer :
    - Visiting a node.
    - Traverse the left sub-tree.
    - Traverse the right sub-tree.
64. **Question 199. What Are The Different Binary Tree Traversal Techniques?**
    Answer :
    - Preorder traversal.
    - Inorder traversal.
    - Postorder traversal.
    - Levelorder traversal.
65. **Question 200. What Is Meant By Binary Tree Traversal?**
    Answer :
    Traversing a binary tree means moving through all the nodes in the binary tree, visiting each node in the tree only once.
66. **Question 201. State The Properties Of A Binary Tree?**
    Answer :
    - The maximum number of nodes on level n of a binary tree is $2n-1$, where $n \geq 1$.
    - The maximum number of nodes in a binary tree of height n is $2n-1$, where $n \geq 1$.
    - For any non-empty tree, $nl=nd+1$ where nl is the number of leaf nodes and nd is the number of nodes of degree 2.
67. **Question 202. Define A Right-skewed Binary Tree?**
    Answer :
    A right-skewed binary tree is a tree, which has only right child nodes.
68. **Question 203. Define A Complete Binary Tree?**

Answer :
A complete binary tree is a tree in which every non-leaf node has exactly two children not necessarily to be on the same level.

69. **Question 204. Define A Full Binary Tree ?**
Answer :
A full binary tree is a tree in which all the leaves are on the same level and every non-leaf node has exactly two children.

70. **Question 205. Define Non-terminal Nodes In A Tree?**
Answer :
All intermediate nodes that traverse the given tree from its root node to the terminal nodes are referred as non-terminal nodes.

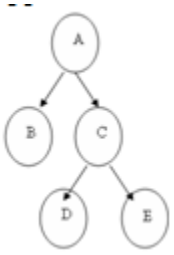
71. **Question 206. Define Terminal Nodes In A Tree?**
Answer :
A node that has no children is called a terminal node. It is also referred to as leaf node.

72. **Question 207. Define A Path In A Tree?**
Answer :
A path in a tree is a sequence of distinct nodes in which successive nodes are connected by edges in the tree.

73. **Question 208. Define A Binary Tree?**
Answer :
A binary tree is a finite set of nodes which is either empty or consists of a root and two disjoint binary trees called the left sub-tree and right sub-tree.

74. **Question 209. Define Forest?**
Answer :
A tree may be defined as a forest in which only a single node (root) has no predecessors. Any forest consists of a collection of trees.

75. **Question 210. What Do You Mean By Level Of The Tree?**
Answer :
The root node is always considered at level zero, then its adjacent children are supposed to be at level 1 and so on.

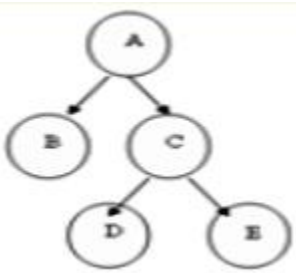


Here, node A is at level 0, nodes B and C are at level 1 and nodes D and E are at level 2.

76. **Question 211. Define Depth And Height Of A Tree?**
Answer :
The depth of the tree is the depth of the deepest leaf. The height of the tree is equal to the height of the root. Always depth of the tree is equal to height of the tree.

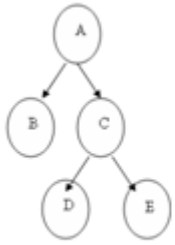
77. **Question 212. Define Depth And Height Of A Node?**
Answer :
For any node ni, the depth of ni is the length of the unique path from the root to ni. The height of ni is the length of the longest path from ni to a leaf.

78. **Question 213. Define Parent Node?**
Answer :
The node which is having further sub-branches is called the parent node of those sub-branches.



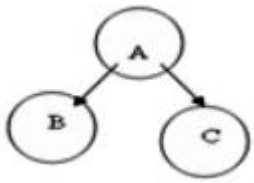
Here C is the parent node of D and E.

79. **Question 214. Define Internal Nodes?**
Answer :
The nodes other than the root and the leaves are called internal nodes.



Here, C is the internal node.

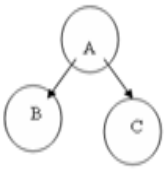
80. **Question 215. Define Leaves?**
Answer :
These are the terminal nodes of the tree. The nodes with degree 0 are always the leaves.



Here C and B are the leave nodes.

81. **Question 216. Define Degree Of The Node?**
Answer :

The total number of sub-trees attached to that node is called the degree of the node.

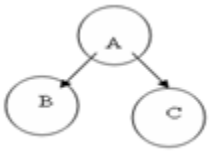


For node A, the degree is 2 and for B and C, the degree is 0.

82. **Question 217. Define Root?**

Answer :

This is the unique node in the tree to which further sub-trees are attached.



Here, A is the root.

83. **Question 218. Define A Tree?**

Answer :

A tree is a collection of nodes. The collection can be empty; otherwise, a tree consists of a distinguished node r, called the root, and zero or more nonempty (sub) trees T1, T2,…,Tk, each of whose roots are connected by a directed edge from r.

84. **Question 219. Why We Need Cursor Implementation Of Linked Lists?**

Answer :

Many languages such as BASIC and FORTRAN do not support pointers. If linked lists are required and pointers are not available, then an alternative implementation must be used known as cursor implementation.

85. **Question 220. List The Applications Of Queues?**

Answer :

- Jobs submitted to printer
- Real life line
- Calls to large companies
- Access to limited resources in Universities
- Accessing files from file server

86. **Question 221. List The Applications Of Stacks?**

Answer :

- Towers of Hanoi
- Reversing a string
- Balanced parenthesis
- Recursion using stack
- Evaluation of arithmetic expressions

87. **Question 222. What Are The Types Of Queues?**

Answer :

- Linear Queues – The queue has two ends, the front end and the rear end. The rear end is where we insert elements and front end is where we delete elements. We can traverse in a linear queue in only one direction ie) from front to rear.
- Circular Queues – Another form of linear queue in which the last position is connected to the first position of the list. The circular queue is similar to linear queue has two ends, the front end and the rear end. The rear end is where we insert elements and front end is where

we delete elements. We can traverse in a circular queue in only one direction ie) from front to rear.
    - Double-Ended-Queue – Another form of queue in which insertions and deletions are made at both the front and rear ends of the queue.

88. **Question 223. What Are The Objectives Of Studying Data Structures?**
    Answer :
    - To identify and create useful mathematical entities and operations to determine what classes of problems can be solved using these entities and operations.
    - To determine the representation of these abstract entities and to implement the abstract operations on these concrete representation.
89. **Question 224. State The Difference Between Persistent And Ephemeral Data Structure?**
    Answer :
    Persistent data structures are the data structures which retain their previous state and modifications can be done by performing certain operations on it. Eg) Stack Ephemeral data structures are the data structures which cannot retain its previous state. Eg) Queues.
90. **Question 225. State The Difference Between Primitive And Non-primitive Data Types?**
    Answer :
    Primitive data types are the fundamental data types. Eg) int, float, double, char Non-primitive data types are user defined data types. Eg) Structure, Union and enumerated data types.
91. **Question 226. What Are The Advantages Of Modularity?**
    Answer :
    - It is much easier to debug small routines than large routines
    - It is easier for several people to work on a modular program simultaneously
    - A well-written modular program places certain dependencies in only one routine, making changes easier
92. **Question 227. Define An Abstract Data Type (adt)?**
    Answer :
    An abstract data type is a set of operations. ADTs are mathematical abstractions; now here in an ADT's definition is there any mention of how the set of operations is implemented. Objects such as lists, sets and graphs, along with their operations can be viewed as abstract data types.
93. **Question 228. Define Data Type And What Are The Types Of Data Type?**
    Answer :
    Data type refers to the kinds of data that variables may hold in the programming language. Eg) int, float, char, double – C

    The following are the types of data type:
    - Built in data type- int, float, char, double which are defined by programming language itself
    - User defined data type- Using the set of built in data types user can define their own data type

    Eg: typedef struct student

```
{
int roll;
char name;
}S;
```

```
S s1;
```

Where S is a tag for user defined data type which defines the structure student and s1 is a variable of data type S.

94. **Question 229. Difference Between Abstract Data Type, Data Type And Data Structure?**
Answer :
    - An Abstract data type is the specification of the data type which specifies the logical and mathematical model of the data type.
    - A data type is the implementation of an abstract data type.
    - Data structure refers to the collection of computer variables that are connected in some specific manner.

    i.e) Data type has its root in the abstract data type and a data structure comprises a set of computer variables of same or different data types.

95. **Question 230. State The Difference Between Queues And Linked Lists?**
Answer :
The difference between queues and linked lists is that insertions and deletions may occur anywhere in the linked list, but in queues insertions can be made only in the rear end and deletions can be made only in the front end.