We can classify Data Structures into two categories:

1. Primitive Data Structure
2. Non-Primitive Data Structure

The following figure shows the different classifications of Data Structures.

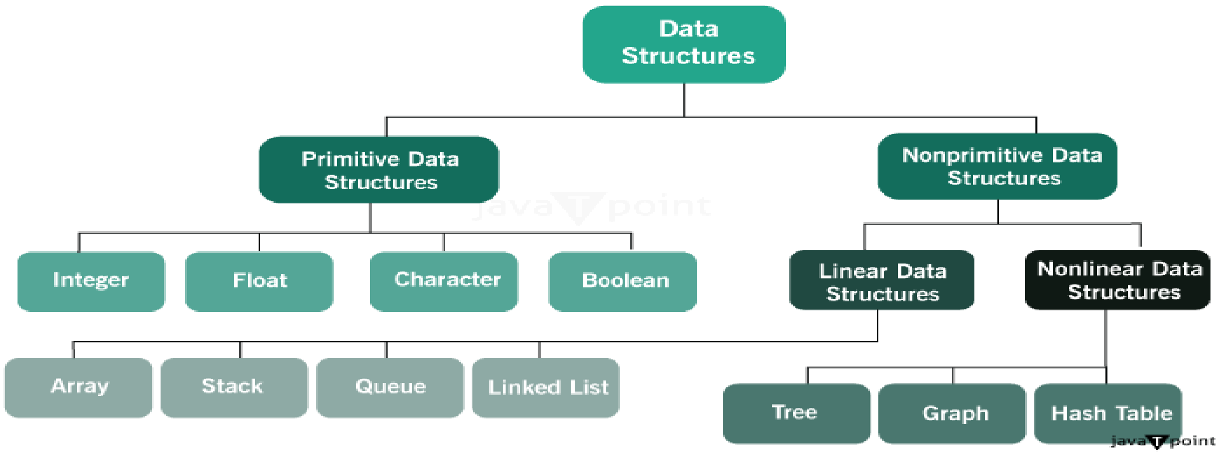

Figure 2: Classifications of Data Structures

Basic Operations of Data Structures

In the following section, we will discuss the different types of operations that we can perform to manipulate data in every data structure:

1. **Traversal:** Traversing a data structure means accessing each data element exactly once so it can be administered. For example, traversing is required while printing the names of all the employees in a department.
2. **Search:** Search is another data structure operation which means to find the location of one or more data elements that meet certain constraints. Such a data element may or may not be present in the given set of data elements. For example, we can use the search operation to find the names of all the employees who have the experience of more than 5 years.
3. **Insertion:** Insertion means inserting or adding new data elements to the collection. For example, we can use the insertion operation to add the details of a new employee the company has recently hired.
4. **Deletion:** Deletion means to remove or delete a specific data element from the given list of data elements. For example, we can use the deleting operation to delete the name of an employee who has left the job.
5. **Sorting:** Sorting means to arrange the data elements in either Ascending or Descending order depending on the type of application. For example, we can use the sorting operation to arrange the names of employees in a department in alphabetical order or estimate the top three performers of the month by arranging the performance of the employees in descending order and extracting the details of the top three.
6. **Merge:** Merge means to combine data elements of two sorted lists in order to form a single list of sorted data elements.
7. **Create:** Create is an operation used to reserve memory for the data elements of the program. We can perform this operation using a declaration statement. The creation of data structure can take place either during the following:
    1. Compile-time

2. Run-time
   For example, the **malloc()** function is used in C Language to create data structure.

8. **Selection:** Selection means selecting a particular data from the available data. We can select any particular data by specifying conditions inside the loop.
9. **Update:** The Update operation allows us to update or modify the data in the data structure. We can also update any particular data by specifying some conditions inside the loop, like the Selection operation.
10. **Splitting:** The Splitting operation allows us to divide data into various subparts decreasing the overall process completion time.

1. What is Data Structure?
arrow_forward
A data structure is a model for organizing and storing data. Stacks, Queue, Linked Lists, and Trees are the examples of different data structures. Data structures are classified as either linear or non-linear:

- A data structure is said to be linear if only one element can be accessed from an element directly. Eg: Stacks, Lists, Queues.
- A data structure is non-linear if more than one elements can be accessed from an element directly. Eg: Trees, Graphs, Heap.

2. Define ADT?
arrow_forward
An abstract data type is a set of objects together with a set of operations. Abstract data types are mathematical abstraction. Objects such as lists, sets, graphs, trees can be viewed as abstract data types.

3. What do you mean by LIFO and FIFO?
arrow_forward
LIFO stands for 'Last In First Out', it says how the data to be stored and retrieved. It means the data or element which was stored last should come out first. Stack follows LIFO.
arrow_forward
FIFO stands for 'First In First Out', here the data(or element) which was stored first should be the one to come out first. It is implemented in Queue.

4. What is Stack?
arrow_forward
A stack is a list of elements in which insertion and deletions can take place only at one end, this end is called as stack 'top'. Only top element can be accessed. A stack data structure has LIFO (Last In First Out) property. A stack is an example of linear data structure.

5. What operations can be done on Stack?
arrow_forward
The following operations can be performed on stack:

- **Push** operation inserts new element into stack.
- **Pop** operation deletes the top element from the stack.
- **Peek** returns or give the top element without deleting it.
- **Isempty()** operation returns whether stack is empty or not.

6. List some applications of stack?
arrow_forward
Some well-known applications of stack are as follow:

1. Infix to Postfix conversion.
2. Evaluatiion of postfix expression.
3. Check for balanced parantheses in an expression.
4. Stack is very important data structure being used to implement function calls efficiently.

5. Parsing
6. Simulation of recursion function.
7. String reverse using stack.

7. What is a Queue?
arrow_forward
A Queue is a particular data structure in which the elements in the collection are kept in order. Unlike Stack, queue is opened at both end. Queue follows 'FIFO' method where element is inserted from rear end and deleted or removed from front end.

8. What operations can be done on Queue?
arrow_forward
The following operations can be performed on queue:

- **Enqueue** operation inserts new element in rear of the list.
- **Dequeue** operation deletes the front element from the list.
- **Isempty()** operation checks whether queue is empty or not.

9. Where do we use Queue?
arrow_forward
Some well-known applications of queue are as follow:

1. For findind level order traversal efficiently.
2. For implementing BFS efficiently.
3. For implementing Kruskal algorithm efficiently.

10. What is Binary Tree?
arrow_forward
A binary tree is a 2-ary tree in which each node(N) has atmost 2 children (either 0 or 1). The node with 2 children are called internal nodes, and the nodes with 0 children are called external nodes or leaf nodes.

11. What is Complete Binary Tree?
arrow_forward
A Binary tree is complete binary tree if all levels are completely filled except possibly the last/lowest level are full and in the last/lowest level all the items are on the left.

12. What is Perfect Balanced Binary Tree?
arrow_forward
A perfect balanced binary tree is a binary tree where each node has same number of nodes in both subtrees.

13. Define Height in a tree?
arrow_forward
Height is a general measured as number of edges from the root to deepest node in the tree.

14. What is tree traversal?
arrow_forward
Traversal is used to visit each node in the tree exactly once. A full traversal of a binary tree gives a linear ordering of the data in the tree. There are 3 different type of tree traversal:

1. Inorder Traversal
2. Preorder Traversal
3. Postorder Traversal

15. How does Inorder Traversal work?

1. Traverse the left sub tree of the root node R in inorder.
2. Visit the root node R.
3. Traverse the right sub tree of the root node R in inorder.

16. How does Preorder Traversal work?

1. Traverse the left sub tree of the root node R in preorder.

2. Traverse the right sub tree of the root node R in preorder.
3. Visit the root node R.

17. How does Postorder Traversal work?

1. Visit the root node R.
2. Traverse the left sub tree of the root node R in postorder.
3. Traverse the right sub tree of the root node R in postorder.

18. What is a Binary Search Tree?

arrow_forward

A BST(Binary Search Tree) is a binary tree in which each node satisfies search property. Each node's key value must be greater than all values in its left subtree and must be less than all values in its right subtree.

What is a bubble sort?

arrow_forward

Bubble sort is a simple sorting algorithm, it works by repeatedly stepping through the list to be sorted comparing each pair of adjacent items and swapping if they are in the worng order.

arrow_forward

First pass bubbles out the largest element and places it in the last position and second pass places the second largest element in the second last position and so on. Thus in the last pass smallest element is placed in the first position.

arrow_forward

The running time is the total number of comparisons that is n(n-1)/2 which implies O(n^2) time complexity.

24. What is Insertion Sort?

arrow_forward

Insertion sort is a simple comparison sorting algorithm, every iteration of insertion sort removes an element from the input data and insert it into the correct position in the already sorted list until no input element remains.

arrow_forward

The running time is the total number of comparisons that is n(n-1)/2 which implies O(n^2) time complexity.

25. What is Selection Sort?

arrow_forward

Insertion sort is a simple comparison sorting algorithm, the algorithm works as follows:

1. Find the minimum value in the list.
2. Swap it with the minimum value in the first position.
3. Repeat the steps above for the remainder of the list (starting at the second position and advancing each time).

arrow_forward

The running time is the total number of comparisons that is n(n-1)/2 which implies O(n^2) time complexity.

26. What is Merge Sort?

arrow_forward

Merge sort is an O(nlogn) comparison-based divide and conquer sorting algorithm, it works as follows:

1. If the list is of length 0 or 1, then it is already sorted.
2. Divide the unsorted list into two sub lists of about half the size.
3. Sort each sublist recursively by re-applying merge sort algorithm.
4. Merge the two sublists back into one sorted list.

arrow_forward

If the running time of merge sort for a list of length n is T(n) then the recurrence relation is T(n) = 2T(n/2) + n, thus after simplifying the recurrence relation T(n) = O(nlogn).

27. What is Heap Sort?

arrow_forward

Heap Sort is a comparison-based sorting algorithm which is much more efficient version of selection sort, it works by determining the largest (or smallest) element of the list, placing that at the end (or beginning)of the list, then continuing with the rest of the list, but accomplishes this task efficiently by using a data structure called a heap. Once the data list has been made into a heap, the root node is gurantedd to be the largest (or smallest) element.

28. What is Quick Sort?

arrow_forward

Quick sort sorts by employing a divide and conquer strategy to divide a list into two sub-lists. The steps are:

1. Pick an element, called a pivot, from the list.
2. Reorder the list so the elements which are less than the pivot come before the pivot and the elements greater than pivot come after it. After this partitioning the pivot is in its final positon.
3. Recursively sort the sub-list of lesser elements and the sub-list of greater elements.

arrow_forward

The running time complexity for worst case is O(n^2) and for best and average case it is same i.e., O(nlogn).

29. What is a Graph?

arrow_forward

A graph is a pair of sets (V,E), where V is the sets of vertices and E is the set of edges connecting the pair of vertices.

30. What is a Directed and Undirected Graph?

arrow_forward

A graph is directed if each edge of graph has a direction between vertices.

arrow_forward

A graph is undirected if there are no direction between vertices.

31. What is a Connected Graph?

arrow_forward

A 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.

32. What is a Complete Graph?

arrow_forward

A complete graph is a graph in which every vertex is connected to every other vertex. That means each vertex's degree in undirected graph is n-1.

33. What is a Acyclic Graph?

arrow_forward

An acyclic graph is a graph which does not have cycles.

34. How are graph represented?

arrow_forward

A graph can be represented in two forms 'Adjacency matrix' and 'Adjacency list'.

arrow_forward

Adjacency matrix is a two dimensional arrays where for each edge, (u,v) A[u,v] = true otherwise it will be false. The space complexity to represent a graph using this is O($|V|^2$).

arrow_forward

Adjacency list are used where each list stores adjacent vertices of the corresponding vertex. The space complexity to represent a graph using this is O(|V|+|E|).

35. What is a Spanning Tree?

arrow_forward

A Spanning tree is a graph which must include every vertex and if the graph contain n vertices, spanning tree must contain exactly (n-1) edges and is a subgraph of G.

36. What is a Minimum Spanning Tree?

arrow_forward

A Minimum Spanning tree is a spanning tree such that the summ of all the weights of edges in spanning tree is minimum.