**Table of Contents**

* [Data Structure Interview Questions](https://hackr.io/blog/data-structure-interview-questions#Data_Structure_Interview_Questions)
  + [Question: What do you understand by a data structure?](https://hackr.io/blog/data-structure-interview-questions#Question_What_do_you_understand_by_a_data_structure)
  + [Question: How does a linear data structure differ from a non-linear data structure?](https://hackr.io/blog/data-structure-interview-questions#Question_How_does_a_linear_data_structure_differ_from_a_non-linear_data_structure)
  + [Question: Please enumerate the various operations that can be performed on a data structure.](https://hackr.io/blog/data-structure-interview-questions#Question_Please_enumerate_the_various_operations_that_can_be_performed_on_a_data_structure)
  + [Question: Can you tell which data structures are used for BFS and DFS of a graph?](https://hackr.io/blog/data-structure-interview-questions#Question_Can_you_tell_which_data_structures_are_used_for_BFS_and_DFS_of_a_graph)
  + [Question: Please explain stack and also mention some of its important applications.](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_stack_and_also_mention_some_of_its_important_applications)
  + [Question: What is a queue? How is it different from a stack?](https://hackr.io/blog/data-structure-interview-questions#Question_What_is_a_queue_How_is_it_different_from_a_stack)
  + [Question: What do you understand by a binary search? What is the best scenario of using it?](https://hackr.io/blog/data-structure-interview-questions#Question_What_do_you_understand_by_a_binary_search_What_is_the_best_scenario_of_using_it)
  + [Question: Could you explain how to reference all the elements in a one-dimension array?](https://hackr.io/blog/data-structure-interview-questions#Question_Could_you_explain_how_to_reference_all_the_elements_in_a_one-dimension_array)
  + [Question: Please explain what do you understand by FIFO and LIFO?](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_what_do_you_understand_by_FIFO_and_LIFO)
  + [Question: Do you know how does dynamic memory allocation help in managing data?](https://hackr.io/blog/data-structure-interview-questions#Question_Do_you_know_how_does_dynamic_memory_allocation_help_in_managing_data)
  + [Question: What is the difference between NULL and VOID?](https://hackr.io/blog/data-structure-interview-questions#Question_What_is_the_difference_between_NULL_and_VOID)
  + [Question: How does a POP operation differ from a PUSH operation?](https://hackr.io/blog/data-structure-interview-questions#Question_How_does_a_POP_operation_differ_from_a_PUSH_operation)
  + [Question: Could you explain how does variable declaration affect memory allocation?](https://hackr.io/blog/data-structure-interview-questions#Question_Could_you_explain_how_does_variable_declaration_affect_memory_allocation)
  + [Question: Please explain the concept of data abstraction.](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_the_concept_of_data_abstraction)
  + [Question: How will you insert a new item in a binary search tree?](https://hackr.io/blog/data-structure-interview-questions#Question_How_will_you_insert_a_new_item_in_a_binary_search_tree)
  + [Question: Could you explain how does the selection sort work on an array?](https://hackr.io/blog/data-structure-interview-questions#Question_Could_you_explain_how_does_the_selection_sort_work_on_an_array)
  + [Question: Do you know how the memory is affected by signed and unsigned numbers?](https://hackr.io/blog/data-structure-interview-questions#Question_Do_you_know_how_the_memory_is_affected_by_signed_and_unsigned_numbers)
  + [Question: Does all declaration statements result in a fixed memory reservation?](https://hackr.io/blog/data-structure-interview-questions#Question_Does_all_declaration_statements_result_in_a_fixed_memory_reservation)
  + [Question: How does an array differ from a stack?](https://hackr.io/blog/data-structure-interview-questions#Question_How_does_an_array_differ_from_a_stack)
  + [Question: What do you understand by an AVL tree?](https://hackr.io/blog/data-structure-interview-questions#Question_What_do_you_understand_by_an_AVL_tree)
  + [Question: Please explain how does an Array differ from a Linked List?](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_how_does_an_Array_differ_from_a_Linked_List)
  + [Question: What do you understand by Infix, Prefix, and Postfix notations?](https://hackr.io/blog/data-structure-interview-questions#Question_What_do_you_understand_by_Infix_Prefix_and_Postfix_notations)
  + [Question: Please explain the Linked List and its various types.](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_the_Linked_List_and_its_various_types)
  + [Question: How will you implement a stack using queue and vice-versa?](https://hackr.io/blog/data-structure-interview-questions#Question_How_will_you_implement_a_stack_using_queue_and_vice-versa)
  + [Question: Which data structures are used for implementing LRU cache?](https://hackr.io/blog/data-structure-interview-questions#Question_Which_data_structures_are_used_for_implementing_LRU_cache)
  + [Question: Could you give a brief explanation of the various approaches for developing algorithms?](https://hackr.io/blog/data-structure-interview-questions#Question_Could_you_give_a_brief_explanation_of_the_various_approaches_for_developing_algorithms)
  + [Question: Please enumerate some examples of greedy and divide and conquer algorithms.](https://hackr.io/blog/data-structure-interview-questions#Question_Please_enumerate_some_examples_of_greedy_and_divide_and_conquer_algorithms)
  + [Question: How does insertion sort differ from selection sort?](https://hackr.io/blog/data-structure-interview-questions#Question_How_does_insertion_sort_differ_from_selection_sort)
  + [Question: What do you understand by shell sort?](https://hackr.io/blog/data-structure-interview-questions#Question_What_do_you_understand_by_shell_sort)
  + [Question: Can you explain tree traversal?](https://hackr.io/blog/data-structure-interview-questions#Question_Can_you_explain_tree_traversal)
  + [Question: Please explain a spanning tree. What is the maximum number of spanning trees a graph can have?](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_a_spanning_tree_What_is_the_maximum_number_of_spanning_trees_a_graph_can_have)
  + [Question: How does the Kruskal’s Algorithm work?](https://hackr.io/blog/data-structure-interview-questions#Question_How_does_the_Kruskals_Algorithm_work)
  + [Question: What do you understand by Heap in data structure?](https://hackr.io/blog/data-structure-interview-questions#Question_What_do_you_understand_by_Heap_in_data_structure)
  + [Question: Please explain recursion.](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_recursion)
  + [Question: Can you explain the Tower of Hanoi problem?](https://hackr.io/blog/data-structure-interview-questions#Question_Can_you_explain_the_Tower_of_Hanoi_problem)
  + [Question: How do the BFS (Breadth First Search) and DFS (Depth First Search) algorithms work?](https://hackr.io/blog/data-structure-interview-questions#Question_How_do_the_BFS_Breadth_First_Search_and_DFS_Depth_First_Search_algorithms_work)
  + [Question: What do you understand by hashing?](https://hackr.io/blog/data-structure-interview-questions#Question_What_do_you_understand_by_hashing)
  + [Question: Please explain an MST (Minimum Spanning Tree). Also, explain how does Prim’s algorithm find a minimum spanning tree.](https://hackr.io/blog/data-structure-interview-questions#Question_Please_explain_an_MST_Minimum_Spanning_Tree_Also_explain_how_does_Prims_algorithm_find_a_minimum_spanning_tree)
  + [Question: Can you explain the interpolation search technique?](https://hackr.io/blog/data-structure-interview-questions#Question_Can_you_explain_the_interpolation_search_technique)
  + [Question: How will you check whether the given Binary Tree is BST or not?](https://hackr.io/blog/data-structure-interview-questions#Question_How_will_you_check_whether_the_given_Binary_Tree_is_BST_or_not)

**Spread the Knowledge**

A data structure can be any organization, management, and storage format of data that allows efficient access and modification. It is a collection of data values, relationships amongst them, and the various functions or operations that can be applied to the data.

Data structures is a foundational concept of programming which is immensely utilized in algorithm design. Hence, it is important for any programmer, irrespective of the programming language, to have a good understanding of data structures.

**Data Structure Interview Questions**

Any programming language interview can have a few or many questions based around data structures. Here are the most important data structure interview questions with their respective answers for you:

**Question**: **What do you understand by a data structure?**

**Answer**: A data structure offers a convenient way of organizing as well as manipulating the data. Simply put, it allows the data to be used in an effective manner. There is a galore of data structures and each of them is suitable for a distinct set of applications.

For instance, compiler implementations use hash tables for looking up identifiers. Similarly, B-trees are suitable for the implementation of databases. Data structures are virtually applied to all areas relying on data. Some of the most important ones are:

* Artificial intelligence
* Compiler design
* Database management
* [Graphics](https://hackr.io/design)
* Numerical analysis
* Operating system
* Statistical analysis

**Question**: **How does a linear data structure differ from a non-linear data structure?**

**Answer**: If the elements of a data structure form a sequence or a linear list then it is called a linear data structure. On the other hand, non-linear data structures are those in which the traversal of nodes is done in a non-linear way.

Arrays, linked lists, stacks, and queues are examples of linear data structures, while graphs and trees are those of non-linear data structures.

**Question**: **Please enumerate the various operations that can be performed on a data structure.**

**Answer**: Following are the various operations that can be performed on a data structure:

* **Deletion –** Deleting an existing element from the data structure
* **Insertion –** Adding a new element to the data structure
* **Searching –** Find the location of an element, if it exists, in the data structure
* **Sorting –** Arranging elements of the data structure in:
  + Ascending or descending order for numerical data
  + Dictionary order for alphanumeric data
* **Traversal –** Accessing each element of the data structure once for processing

**Question**: **Can you tell which data structures are used for BFS and DFS of a graph?**

**Answer**: BFS (Breadth First Search) of a graph uses a queue. Although DFS (Depth First Search) of a graph makes use of a stack, it can also be implemented using recursion that uses function call stack.

**Question**: **Please explain stack and also mention some of its important applications.**

**Answer**: Stack is a linear data structure that follows either LIFO (Last In First Out) or FILO (First In Last Out) approach for accessing elements. Push, pop, and peek are the basic operations of a stack.

Some notable applications of a stack are:

* Check for balanced parentheses in an expression
* Evaluation of a postfix expression
* Implement two stacks in an array
* Infix to postfix conversion
* Reverse a string

**Question**: **What is a queue? How is it different from a stack?**

**Answer**: A queue is a form of linear structure that follows the FIFO (First In First Out) approach for accessing elements. Dequeue, enqueue, front, and rear are basic operations on a queue. Like a stack, a queue can be implemented using arrays and linked lists.

In a stack, the item that is most recently added is removed first. Contrary to this, the item least recently added is removed first in case of a queue.

**Question**: **What do you understand by a binary search? What is the best scenario of using it?**

**Answer**: A binary search is an algorithm that starts with searching in the middle element. If the middle element is not the target element then it further checks whether to continue searching the lower half or the higher half. The process continues until the target element is found.

The binary search works best when applied to a list with sorted or ordered elements.

**Question**: **Could you explain how to reference all the elements in a one-dimension array?**

**A**: We can reference all the elements in a one-dimension array using an indexed loop. The counter runs from 0 to the maximum array size, say n, minus one. All elements of the one-dimension array are referenced in sequence by using the loop counter as the array subscript.

**Question**: **Please explain what do you understand by FIFO and LIFO?**

**Answer**: Both FIFO and LIFO are approaches to accessing, storing, and retrieving elements from a data structure. LIFO stands for Last In First Out. In this approach, recently stored data is the one to be extracted first.

FIFO is a contraction for First In First Out. Following this approach, the data that is stored the least recently will be extracted first.

**Question**: **Do you know how does dynamic memory allocation help in managing data?**

**Answer**: Dynamic memory allocation helps in storing simple structured data types. Moreover, it can combine separately allocated structured blocks for forming composite structures that contract and expand as required.

**Question**: **What is the difference between NULL and VOID?**

**Answer**: While NULL is a value, VOID is a data type identifier. A variable assigned with a NULL value represents an empty value. The VOID is used for identifying pointers having no initial size.

**Question**: **How does a POP operation differ from a PUSH operation?**

**Answer**: Both PUSH and POP operations pertain to a stack. Data is added to the stack using the PUSH operation, while it is retrieved using the POP operation.

**Question**: **Could you explain how does variable declaration affect memory allocation?**

**Answer**: The total amount of memory to be allocated or reserved in the case of a variable declaration depends on the data type used. For instance, declaring an integer type variable reserves 4 bytes of memory space while declaring a double variable reserve 8 bytes of the available memory.

**Question**: **Please explain the concept of data abstraction.**

**Answer**: Data abstraction helps in dividing complex data problems into smaller, easy-to-manage parts. It starts with specifying all the involved data objects and the various operations to be performed on the same without stressing too much on the way data is stored.

**Question**: **How will you insert a new item in a binary search tree?**

**Answer**: As a binary search tree doesn’t allow for duplicates, the new item to be inserted must be unique. Assuming it is, we will proceed with checking whether the tree is empty or not. If it is empty, then the new item will be inserted in the root node.

However, if the tree is non-empty then we will refer to the key of the new item. When it is smaller than the root item’s key, the new item will be added to the left subtree. If the new item’s key is bigger than the root item’s key, then the new item is inserted into the right subtree.

**Question**: **Could you explain how does the selection sort work on an array?**

**Answer**: The selection sort begins with finding the smallest element. It is switched with the element present at subscript 0. Next, the smallest element in the remaining subarray is located and switched with the element residing in the subscript 1.

The aforementioned process is repeated until the biggest element is placed at the subscript n-1, where n represents the size of the given array.

**Question**: **Do you know how the memory is affected by signed and unsigned numbers?**

**Answer**: For signed numbers, the first bit is reserved for indicating whether the number is positive or negative. Hence, it has one bit less for storing the value. Unlike signed numbers, unsigned numbers have all the bits available for storing the number.

The effect of the aforementioned can be seen in the value range available to signed and unsigned numbers. While an unsigned 8-bit number can have a range of 0 to 255, an 8-bit signed number has a range varying from -128 to 127.

**Question**: **Does all declaration statements result in a fixed memory reservation?**

**Answer**: Except for pointers, all declaration statements result in a fixed memory reservation. Instead of allocating memory for storing data, a pointer declaration results into allocating memory for storing the address of the pointer variable.

For pointers, actual memory allocation for the data happens during the runtime.

**Question**: **How does an array differ from a stack?**

**Answer**: A stack follows the LIFO approach. This means that data manipulation follows a specific sequence where the latest data element is the one to be retrieved first.

Unlike a stack, an array doesn’t follow any particular sequence for adding or retrieving data. Adding or retrieving an element in an array is done by referring to the array index.

**Question**: **What do you understand by an AVL tree?**

**Answer**: An AVL tree is a type of BST (Binary Search Tree), which is always in a partially-balanced state. The measure of the balance is given by the difference of the heights of the subtrees from the root node of the AVL tree.

**Question: Please explain how does an Array differ from a Linked List?**

**Answer**: Following are the various differences between an array and a linked list:

* **Additional Memory –** For each element belonging to a linked list, extra memory space is required for storing the pointer. Arrays have no such requirement
* **Cache –** In comparison to linked lists, arrays have better cache locality, which can significantly enhance the performance in various scenarios
* **Insertion and Deletion –** It is easy to add or delete elements in a linked list. Inserting and deleting elements for an array is comparatively expensive
* **Random Access –** Linked lists do not allow random access, while arrays do
* **Size –** While the size of an array is fixed, the size of a linked list is dynamic

**Question: What do you understand by Infix, Prefix, and Postfix notations?**

**Answer**:

* **Infix Notation –**Operators are written between the operands. This is the standard way of writing expressions. For example, A \* (B + C) / D
* **Postfix Notation/Reverse Polish Notation –** Operators are written after the operands, hence the name. For instance, A B C + \* D /
* **Prefix Notation/Polish Notation –** Operators are written before the operands. / \* A + B C D is the prefix notation equivalent of the aforementioned postfix notation example

**Question: Please explain the Linked List and its various types.**

**Answer**: In a linked list, each element is a distinct object. Like arrays, linked lists are a linear type of [data structures](https://hackr.io/blog/python-data-structures). In addition to data, every element of a linked list comprises a reference to the next element. Various types of linked lists are:

* **Singly Linked List –** Each node stores the address or reference of the next node in the linked list, leave for the last node that stores NULL
* **Doubly Linked List –** Each node keeps two references. One point to the next node and the other points to the previous node
* **Circular Linked List –** In this type of linked list, all nodes are connected to form a circle. Hence, there is no NULL at the end. A circular linked list can either be a single circular linked list or a double circular linked list

**Question: How will you implement a stack using queue and vice-versa?**

**Answer**: It is possible to implement a stack using two queues. Further, there are two options; either to make the push operation costly or the pop operation costly.

A queue can also be implemented with two stacks. Moreover, there are two options; either to make the enQueue operation costly or the deQueue operation costly.

**Question: Which data structures are used for implementing LRU cache?**

**Answer**: By organizing items in order of use, a Least Recently Used or LRU cache allows quick identification of an item that hasn’t been put to use for the longest time. Two data structures are used for implementing an LRU cache:

* **Queue –**Implemented using a doubly linked list. The maximum size of the queue is determined by the total number of frames available i.e. the cache size. While the most recently used pages will be near the rear end of the queue, the least recently pages will be near the queue’s front end
* **Hashmap –** Having page number as the key along with the address of the corresponding queue node as the value

**Question: Could you give a brief explanation of the various approaches for developing algorithms?**

**Answer**: There are 3 main approaches to developing algorithms:

* **Divide and Conquer –**Involves dividing the entire problem into a number of subproblems and then solving each of them independently
* **Dynamic Programming –** Identical to the divide and conquer approach with the exception that all sub-problems are solved together
* **Greedy Approach –** Finds a solution by choosing the next best option

**Question: Please enumerate some examples of greedy and divide and conquer algorithms.**

**Answer**: Some examples of algorithms that follow greedy approach are:

* Dijkstra’s Minimal Spanning Tree
* Graph – Map Coloring
* Graph – Vertex Cover
* Job Scheduling Problem
* Knapsack Problem
* Kruskal’s Minimal Spanning Tree
* Prim’s Minimal Spanning Tree
* Travelling Salesman

Following are some notable instances of the divide and conquer approach:

* Binary Search
* Closest Pair (or Points)
* [Merge Sort](https://hackr.io/blog/merge-sort-in-c)
* Quick Sort
* Strassen’s Matrix Multiplication

**Question: How does insertion sort differ from selection sort?**

**Answer**: Both insertion and selection approaches maintain two sub-lists, sorted and unsorted. Each takes one element from the unsorted sub-list and place it into the sorted sub-list. The distinction between the two sorting processes lies in the treatment of the current element.

Insertion sort takes the current element and places it in the sorted sublist at the appropriate location. Selection sort, on the other hand, searches for the minimum value in the unsorted sub-list and replaces the same with the present element.

**Question: What do you understand by shell sort?**

**Answer**: The shell sort can be understood as a variant of the insertion sort. The approach divides the entire list into smaller sub-lists based on some gap variable. Each sub-list is then sorted using insertion sort.

**Question: Can you explain tree traversal?**

**Answer**: The process for visiting all the nodes of a tree is called tree traversal. It always starts from the root node and there are three ways of doing it:

* In-order Traversal
* Pre-order Traversal
* Post-order Traversal

**Question: Please explain a spanning tree. What is the maximum number of spanning trees a graph can have?**

**Answer**: A spanning tree is a subset of a graph that has all the vertices but with the minimum possible number of edges. Neither a spanning tree can be disconnected and nor does it have cycles.

The maximum number of spanning trees that a graph can have depended on how connected the graph is. A complete undirected graph with n number of nodes can have a maximum of nn-1 number of spanning trees.

**Question: How does the Kruskal’s Algorithm work?**

**Answer**: Kruskal’s algorithm treats a graph as a forest and each node in it as an individual tree. A tree connects to another tree only if it:

* Has the least cost among all the available options
* Does not violate the [MST properties](https://www.quora.com/What-are-some-properties-of-minimum-spanning-trees)

**Question: What do you understand by Heap in data structure?**

**Answer**: A Heap data structure is a special balanced binary tree in which the root node key is compared with its children and accordingly arranged. A Heap data structure can be of two types:

* **Min-Heap –**The parent node has a key value less than its children
* **Max-Heap –** The parent node has a key value greater than its children

**Question: Please explain recursion.**

**Answer**: The ability to allow a function or module to call itself is called recursion. Either a function f calls itself directly or calls another function ‘g’ that in turn calls the function ‘f. The function f is known as the recursive function and it follows the recursive properties:

* **Base criteria –** Where the recursive function stops calling itself
* **Progressive approach –** Where the recursive function tries to meet the base criteria in each iteration

**Question: Can you explain the Tower of Hanoi problem?**

**Answer**: The Tower of Hanoi is a mathematical puzzle that comprises of three tower (or pegs) and more than one ring. Each ring is of varying size and stacked upon one another such that the larger one is beneath the smaller one.

The goal of the Tower of Hanoi problem is to move the tower of the disk from one peg to another without breaking the properties.

**Question: How do the BFS (Breadth First Search) and DFS (Depth First Search) algorithms work?**

**Answer**: The BFS algorithm traverses a graph in the breadthwards motion. It uses a queue to remember the next vertex for starting a search when a dead end occurs in any iteration.

A DFS algorithm traverses a graph in the depthward motion. It uses a stack for remembering the next vertex to start a search when coming across a dead end in an iteration.

**Question: What do you understand by hashing?**

**Answer**: The technique of converting a range of key values into a range of indexes of an array is known as hashing. It is possible to create associative data storage using hash tables where data indices can be found by providing the corresponding key values.

**Question: Please explain an MST (Minimum Spanning Tree). Also, explain how does Prim’s algorithm find a minimum spanning tree.**

**Answer**: An MST or Minimum Spanning Tree is a spanning tree in a weighted graph that has the minimum weight of all the possible spanning trees. Each node is treated as a single tree by [Prim’s algorithm](https://en.wikipedia.org/wiki/Prim's_algorithm) while adding new nodes to the spanning tree from the available graph.

**Question: Can you explain the interpolation search technique?**

**Answer**: The interpolation search technique is an enhanced variant of [binary search](https://hackr.io/blog/binary-search-in-c). It works on the probing position of the required value.

**Question: How will you check whether the given Binary Tree is BST or not?**

**Answer**: Simply do inorder traversal of the given binary tree while keeping track of the previous key value. If the current key value is greater, then continue, otherwise return false. The binary tree is BST if the inorder traversal of the binary tree is sorted.

That sums up the list of the 20 very important data structure interview questions. Looking to further your data structure knowledge? Try these [best data structure tutorials](https://hackr.io/tutorials/learn-data-structures-algorithms?ref=blog-post?ref=blog-post). Good luck!

**People Also Read:**