Data_Structures - Hackveda

(Talk to Live Mentor)

Become a Data Structure and Algorithm Professional - (Beginner - Advanced)

Skill level: Beginner - Advanced

Training fee: INR 15999 only (Topics covered: 192)

Chief Trainer: Mr. Devanshu Shukla

Training Duration: 32 days (3 hrs per day) | 48 days (2 hrs per day) | 96 days (1 hr per day)

Presentation or Examination will be conducted within 32 days from date of training completion.

* Please note examination will be conducted after completion of training.

Maximum examination attempts: 03

Minimum passing marks for certification and placement: 90%

Query Membership: 01 year (Online / Offline)

Spoken Language: English / Hindi

Venue: Hackveda, H-3/60, III Floor, Sector-18, Rohini, Delhi-110089

Contact person: Mr. Yash Sharma, Software Engineer, Hackveda

Contact phone: 011-27297608, +91-9654825370, +91-9891799066

Registration link: Register Now

Hackveda One2One Support Available:
Training session video will be recorded and delivered to students via our Digital Learning platform Hackveda One2One for any time, any where learning and practice.

Join the training at Hackveda 'TODAY' !

Course contents

One-Dimensional Arrays

Working with Arrays

Two-Dimensional Arrays

Multi-Dimensional Arrays

Lesson 1 Test

Strings and Character Arrays

Reading and Writing Strings and Characters

Manipulating Strings

Lesson 2 Test

Declaring Structures

Using Nested Structures

Structures and Arrays

Structures and Functions

Pointing to Structures

Structures and Bit Fields

Lesson 3 Test

Introduction to Data Structures

Introduction to Algorithms

Introduction to Time and Space Complexity

Introduction to Static Arrays

Introduction to Dynamic Arrays

Introduction to Recursion Using Binary Search

Lesson 4 Test

Introduction to the Stack

Introduction to Queues

Introduction to the Linked List

Lesson 5 Test

Introduction to the Binary Search Tree

Binary Search Tree ? Performing a Search

Binary Search Tree ? Inserting Elements

Binary Search Tree ? Deleting Elements

Lesson 6 Test

Introduction to Sorting ? Bubble Sort

Sorting Using Merge Sort

Sorting Using Quicksort

Lesson 7 Test

Representing Graphs ? Adjacency List

Representing Graphs ? Adjacency Matrix

Graph Searching ? Breadth First Search

Graph Searching ? Depth First Search

Graph Sorting ? Topological Sort

Lesson 8 Test

Introduction to Hashed Data Structures

Hashed Data Structures ? The Hash Function

Hashed Data Structures ? Perfect vs Non-Perfect Hashing

Hashed Data Structures ? Handling Collisions

Lesson 9 Test

Exercise: Using Algorithms and Data Structure in C++

Course Test

Introduction to STL Algorithms in C++

Using std-for_each in C++ Lambda Expressions

Introduction to Iterators Using Find in C++

Using std-find_if in C++

Using std-find_end in C++

Using std-find_first_of in C++

Using std-adjacent_find in C++

Using std-count and count_if in C++

Using std-mismatch and equal in C++

Using std-search and search_n in C++

Using Nonmodifying Algorithms with Containers

Lesson 10 Test

Using std-transform in C++

Using Iterators to Copy and Move in C++

Using std-copy and copy_backward in C++

Using std-swap and swap_ranges in C++

Using std-iter_swap in C++

Using std-replace in C++

Using std-fill and fill_n in C++

Using std-generate and generate_n in C++

Using std-remove and remove_if in C++

Using std-unique and unique_copy in C++

Using std-reverse and reverse_copy in C++

Using std-rotate in C++

Using std-partition and stable_partition in C++

Using Modifying Algorithms with Containers

Lesson 11 Test

Exercise: Working with Algorithms in C++

Course Test

Java Primer

1.1: Getting Started

1.2: Classes and Objects

1.3: Strings Wrappers Arrays and Enum Types

1.4: Expressions

1.5: Control Flow

1.6: Simple Input and Output

1.7: An Example Program

1.8: Packages and Imports

1.9: Software Development

Object-Oriented Design

2.1: Goals Principles and Patterns

2.2: Inheritance

2.3: Interfaces and Abstract Classes

2.4: Exceptions

2.5: Casting and Generics

2.6: Nested Classes

2.7: Course Test

Fundamental Data Structures

3.1: Using Arrays

3.2: Singly Linked Lists

3.3: Circularly Linked Lists

3.4: Doubly Linked Lists

3.5: Equivalence Testing

3.6: Cloning Data Structures

3.7: Course Test

Algorithm Analysis

Overview

4.1: Experimental Studies

4.2: The Seven Functions Used in This Book

4.3: Asymptotic Analysis

4.4: Simple Justification Techniques

4.5: Course Test

Recursion

Overview

5.1: Illustrative Examples

5.2: Analyzing Recursive Algorithms

5.3: Further Examples of Recursion

5.4: Designing Recursive Algorithms

5.5: Recursion Run Amok

5.6: Eliminating Tail Recursion

Stacks Queues and Deques

6.1: Stacks

6.2: Queues

6.3: Double-Ended Queues

List and Iterator ADTs

7.1: The List ADT

7.2: Array Lists

7.3: Positional Lists

7.4: Iterators

7.5: The Java Collections Framework

7.6: Sorting a Positional List

7.7: Case Study: Maintaining Access Frequencies

Trees

8.1: General Trees

8.2: Binary Trees

8.3: Implementing Trees

8.4: Tree Traversal Algorithms

8.5: Course Test

Priority Queues

9.1: The Priority Queue Abstract Data Type

9.2: Implementing a Priority Queue

9.3: Heaps

9.4: Sorting with a Priority Queue

9.5: Adaptable Priority Queues

9.6: Course Test

Maps Hash Tables and Skip Lists

10.1: Maps

10.2: Hash Tables

10.3: Sorted Maps

10.4: Skip Lists

10.5: Sets Multisets and Multimaps

10.6: Course Test

Search Trees

11.1: Binary Search Trees

11.2: Balanced Search Trees

11.3: AVL Trees

11.4: Splay Trees

11.5: (2 4) Trees

11.6: Red-Black Trees

11.7: Course Test

Sorting and Selection

12.1: Merge-Sort

12.2: Quick-Sort

12.3: Studying Sorting through an Algorithmic Lens

12.4: Comparing Sorting Algorithms

12.5: Selection

12.6: Course Test

Text Processing

13.1: Abundance of Digitized Text

13.2: Pattern-Matching Algorithms

13.3: Tries

13.4: Text Compression and the Greedy Method

13.5: Dynamic Programming

13.6: Course Test

Graph Algorithms

14.1: Graphs

14.2: Data Structures for Graphs

14.3: Graph Traversals

14.4: Transitive Closure

14.5: Directed Acyclic Graphs

14.6: Shortest Paths

14.7: Minimum Spanning Trees

14.8: Course Test

Memory Management and B-Trees

15.1: Memory Management

15.2: Memory Hierarchies and Caching

15.3: External Searching and B-Trees

15.4: External-Memory Sorting

15.5: Course Test

How to Join

1.) Register your name online at Register Now

2.) Deposit your training fee via IMPS / NEFT / PAYTM / Google Tez / Phone Pe or Cash Deposit at Training Centre

3.) Send snapshot / transaction number via Whatsapp to +91-9654825370 or Email us at admin@hackveda.in

4.) Bill will be generated and sent to your Email ID, Hackveda One2One account details will also be sent via sms and email. You can also collect Hackveda One2One account details from training centre.

Bank Details for IMPS / Paytm to Bank / NEFT / ATM - Cash Deposit

Name: Devanshu Shukla

Account Number: 55142333064

Bank Name: State Bank of India

Branch: Rama Market, Pitampura

IFS Code: SBIN0050403

Pay via PayTM / Google Tez / Phone Pe

9654825370

Optional Pre-requisites

Laptop & Charger, 4GB+ Pendrive, Headphones

Training Centres

Hackveda - H-3/60, III Floor, Sector-18, Rohini, Delhi - 110089