# Data Structures & Algorithms with Real-World Applications

|  |  |  |
| --- | --- | --- |
| DSA | Description | Real-World Example |
| Array | Stores elements in contiguous memory locations. | Image pixels, music samples in MP3 files. |
| Linked List | Nodes linked together where each node stores data and pointer to next. | Music playlist, browser history navigation. |
| Stack | LIFO (Last In First Out) structure. | Undo/Redo in text editors, function call stack in programming. |
| Queue | FIFO (First In First Out) structure. | Printer queue, task scheduling. |
| Deque | Double-ended queue supporting insert/delete at both ends. | Text editor undo/redo with flexibility, browser tabs navigation. |
| Hash Table | Stores key-value pairs with fast lookups. | Password verification, database indexing. |
| Heap | Special tree-based structure for priority handling. | Priority queues in operating systems, Dijkstra’s shortest path. |
| Graph | Collection of nodes connected by edges. | Social networks, Google Maps routing. |
| Binary Tree | Hierarchical data structure with at most two children per node. | File systems, decision-making algorithms. |
| Binary Search Tree | Sorted binary tree for fast search, insert, delete. | Database indexing, auto-suggestions in search engines. |
| Trie | Tree for storing strings efficiently. | Autocomplete, spell checker. |
| Segment Tree | Stores intervals for efficient range queries. | Gaming leaderboards, analytics dashboards. |
| Fenwick Tree | Efficient structure for prefix sums. | Cumulative frequency analysis, financial data analysis. |
| B-Trees | Self-balancing search tree for large data. | Databases, file systems. |
| AVL Tree | Self-balancing binary search tree. | Memory indexing, CPU scheduling. |
| Red-Black Tree | Balanced binary search tree with color properties. | Map and Set in C++ STL, Java TreeMap. |
| DFS (Depth First Search) | Graph traversal algorithm exploring as far as possible. | Maze solving, game AI pathfinding. |
| BFS (Breadth First Search) | Graph traversal exploring level by level. | Shortest path in unweighted graphs, peer-to-peer networks. |
| Dijkstra’s Algorithm | Finds shortest paths in weighted graphs. | GPS navigation, network routing. |
| Bellman-Ford Algorithm | Finds shortest paths, supports negative weights. | Currency arbitrage detection, network analysis. |
| Floyd-Warshall Algorithm | Finds shortest paths between all pairs. | Social network analysis, transportation systems. |
| Kruskal’s Algorithm | Finds Minimum Spanning Tree. | Network design, clustering. |
| Prim’s Algorithm | Another Minimum Spanning Tree approach. | Road construction, cable network design. |
| Merge Sort | Divide and conquer sorting algorithm. | Sorting large datasets, external sorting. |
| Quick Sort | Efficient divide and conquer sorting. | E-commerce price sorting, leaderboard rankings. |
| Heap Sort | Sorting using heap data structure. | Priority task arrangement, job scheduling. |
| Bubble Sort | Simple comparison-based sort. | Teaching sorting basics, small datasets. |
| Insertion Sort | Sorts by inserting elements into correct position. | Card games, small dataset sorting. |
| Selection Sort | Selects minimum and swaps until sorted. | Basic hardware-level sorting. |
| Counting Sort | Non-comparison sort for integers. | Sorting exam scores, age distribution analysis. |
| Radix Sort | Sorts numbers digit by digit. | Sorting phone numbers, large integer datasets. |
| KMP Algorithm | Efficient string matching algorithm. | Text search in editors, plagiarism detection. |
| Rabin-Karp Algorithm | String matching using hashing. | Search engines, detecting plagiarism. |
| Z Algorithm | Pattern matching algorithm. | DNA sequence analysis, search highlighting. |
| Aho-Corasick Algorithm | Multi-pattern string matching. | Spam filtering, search in large texts. |