**Most important type of Algorithms**

**Algorithm:**  
An algorithm is a step-by-step procedure to solve a problem. A good algorithm should be optimized in terms of time and space. Different types of problems require different types of algorithmic-techniques to be solved in the most optimized manner. There are many types of algorithms but the most important and the fundamental algorithms that you must know will be discussed in this article.

**Brute Force Algorithm:**  
This is the most basic and simplest type of algorithm. A Brute Force Algorithm is the straightforward approach to a problem i.e., the first approach that comes to our mind on seeing the problem. More technically it is just like iterating every possibility available to solve that problem.

**For Example:** If there is a lock of **4-digit** PIN. The digits to be chosen from **0-9** then the brute force will be trying all possible combinations one by one like **0001**, **0002**, **0003**, **0004**, and so on until we get the right PIN. In the worst case, it will take **10,000 tries** to find the right combination.

**Recursive Algorithm:**  
This type of algorithm is based on [recursion](https://www.geeksforgeeks.org/recursion/). In recursion, a problem is solved by breaking it into subproblems of the same type and calling own self again and again until the problem is solved with the help of a base condition.

Some common problem that is solved using recursive algorithms are [Factorial of a Number](https://www.geeksforgeeks.org/program-for-factorial-of-a-number/), [Fibonacci Series](https://www.geeksforgeeks.org/program-for-nth-fibonacci-number/), [Tower of Hanoi](https://www.geeksforgeeks.org/c-program-for-tower-of-hanoi/), [DFS for Graph](https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/), etc.

[Divide and Conquer Algorithm](https://www.geeksforgeeks.org/divide-and-conquer-introduction/)**:**  
In Divide and Conquer algorithms, the idea is to solve the problem in two sections, the first section divides the problem into subproblems of the same type. The second section is to solve the smaller problem independently and then add the combined result to produce the final answer to the problem.

Some common problem that is solved using Divide and Conquers Algorithms are [Binary Search](https://www.geeksforgeeks.org/binary-search/), [Merge Sort](https://www.geeksforgeeks.org/merge-sort/), [Quick Sort,](https://www.geeksforgeeks.org/quick-sort/)[Strassen’s Matrix Multiplication](https://www.geeksforgeeks.org/strassens-matrix-multiplication/), etc.

[Dynamic Programming Algorithms](https://www.geeksforgeeks.org/dynamic-programming/)**:**  
This type of algorithm is also known as the [memoization technique](https://www.geeksforgeeks.org/memoization-1d-2d-and-3d/) because in this the idea is to store the previously calculated result to avoid calculating it again and again. In Dynamic Programming, divide the complex problem into smaller [overlapping subproblems](https://www.geeksforgeeks.org/overlapping-subproblems-property-in-dynamic-programming-dp-1/) and storing the result for future use.

The following problems can be solved using Dynamic Programming algorithm [Knapsack Problem](https://www.geeksforgeeks.org/0-1-knapsack-problem-dp-10/), [Weighted Job Scheduling](https://www.geeksforgeeks.org/weighted-job-scheduling/), [Floyd Warshall Algorithm](https://www.geeksforgeeks.org/floyd-warshall-algorithm-dp-16/), [Dijkstra Shortest Path Algorithm](https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/), etc.

[Greedy Algorithm](https://www.geeksforgeeks.org/greedy-algorithms/)**:**  
In the Greedy Algorithm, the solution is built part by part. The decision to choose the next part is done on the basis that it gives the immediate benefit. It never considers the choices that had taken previously.

Some common problems that can be solved through the Greedy Algorithm are [Prim’s Algorithm](https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/), [Kruskal’s Algorithm](https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/), [Huffman Coding](https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/), etc.

[Backtracking Algorithm](https://www.geeksforgeeks.org/backtracking-algorithms/)**:**  
In Backtracking Algorithm, the problem is solved in an incremental way i.e. it is an algorithmic-technique for solving problems recursively by trying to build a solution incrementally, one piece at a time, removing those solutions that fail to satisfy the constraints of the problem at any point of time.

Some common problems that can be solved through the Backtracking Algorithm are [Hamiltonian Cycle](https://www.geeksforgeeks.org/hamiltonian-cycle-backtracking-6/), [M-Coloring Problem](https://www.geeksforgeeks.org/m-coloring-problem-backtracking-5/), [N Queen Problem](https://www.geeksforgeeks.org/n-queen-problem-backtracking-3/), [Rat in Maze Problem](https://www.geeksforgeeks.org/rat-in-a-maze-backtracking-2/), etc.