Sorting lets the algorithm know which half to ignore. If not sorted, the middle value tells you nothing.

Hash tables chain or probe to deal with collisions. If not handled well, collisions cause near linear time.

A min-heap fixes itself by swapping parents and children until order is correct. Both insert and delete use this.

BFS explores level by level. Because all edges count the same, the level is the distance.

Random pivots make quicksort fast most of the time. But a bad pattern makes the splits uneven and slow.

Linked list insertion only moves pointers. Arrays need to shift many items, which is slow.

DP solves each small part once and stores it. Overlap means you save a lot of work.

DFS uses a stack to push deeper nodes first. It changes the path it explores.

Tries store words through a chain of nodes per letter. This helps with fast prefix search.

Balanced BSTs keep operations at log time. If the tree leans, search time grows a lot.