Here is an expanded list that includes a few more sorting algorithms along with their time complexity, space complexity, a brief explanation, and stability:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sorting Algorithm | Time Complexity | Space Complexity | Explanation (Placeholder2) | Stability |
| Bubble Sort | O(n^2) | O(1) | Simple and inefficient, repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order. | Stable |
| Selection Sort | O(n^2) | O(1) | Sort an array by repeatedly finding the minimum element from the unsorted part and putting it at the beginning. | Unstable |
| Insertion Sort | O(n^2) | O(1) | Builds the final sorted array one item at a time. It is much less efficient on large lists than more advanced algorithms. | Stable |
| Merge Sort | O(n log n) | O(n) | A divide and conquer algorithm that divides the input array into two halves, recursively sorts them, and then merges the sorted halves. | Stable |
| Quick Sort | O(n^2) (worst) | O(log n) (average) | Another divide and conquer algorithm that works by selecting a 'pivot' element and partitioning the other elements into two sub-arrays according to whether they are less than or greater than the pivot. | Unstable |
| Heap Sort | O(n log n) | O(1) | Builds a max-heap (or min-heap) to represent the array and repeatedly removes the maximum (or minimum) element to build the sorted array. | Unstable |
| Counting Sort | O(n + k) | O(k) | Assumes that each element in the input is an integer within a specific range. It counts the occurrences of each element and uses that information to reconstruct a sorted sequence. | Stable |
| Radix Sort | O(nk) | O(n + k) | Processes the digits of the numbers in a specific order, from the least significant digit to the most significant digit, or vice versa. | Stable |
| Bucket Sort | O(n^2) (worst) | O(n + k) | Distributes the elements into a number of buckets and then individually sorts each bucket, typically using a different sorting algorithm. | Stable |
| Shell Sort | O(n log^2 n) (worst-case) | O(1) | A variation of insertion sort that allows the exchange of items that are far apart, producing partially sorted arrays that can be efficiently sorted with subsequent passes. | Unstable |
| Cocktail Shaker Sort | O(n^2) | O(1) | A variation of the bubble sort algorithm that sorts the list in both directions, eliminating turtles (small values near the end) and rabbits (large values near the beginning). | Stable |
| Comb Sort | O(n^2) (worst) | O(1) | An improvement over bubble sort that eliminates small values near the end of the list efficiently by using a gap sequence. | Unstable |
| Gnome Sort | O(n^2) | O(1) | A simple sorting algorithm that works by repeatedly swapping adjacent elements if they are in the wrong order, similar to the behavior of a gnome sorting a line of flowerpots. | Stable |