Description for each algorithm:

Bubblesort:

This is the simplest of these algorithms, it is also rather comically inefficient. It works by working its way through elements and swapping adjacent elements if the left one is of higher value than the right one. It goes through all the elements and then keeps iterating through the elements until it makes a whole pass without swapping any elements.

Quicksort:

This genious algorithm is similar in thought to binary searching but for sorting instead. The algorithm splits (whatever it is working with) the array in two, separating the array at a value (the pivot). This is then repeated for the separated halves, and again, and again…etc recursively.

Mergesort:

This algorithm has a time complexity of O(n log n) and this is because the way it works is by splitting the array (or whatever you are working with) in two halves, and then doing this again recursively until the arrays can no longer be broken down (if only one element is part of a list/array then it is sorted). Then the “merge” part comes up, the arrays are merged and when this happens the values of the two (already sorted) arrays are compared. If the values are ordered nothing happens if they are not ordered, then they swap places. This then happens until all the values are merged.