	Best case	Worst case	Average case
Insertion sort			
Selection sort			
Heap sort			
Mergesort			
Quicksort			

Bucket sort

Radix sort:
Input: 478, 537, 9, 721, 3, 38, 123, 67

For each of the following situations, name the best sorting algorithm we studied. (For one or two questions, there may be more than one answer deserving full credit, but you only need to give one answer for each.)

(a) The array is mostly sorted already (a few elements are in the wrong place).

(b) You need an O(n log n) sort even in the worst case and you cannot use any extra space except for a few local variables.

(c) The data to be sorted is too big to fit in memory, so most of it is on disk.

(d) You have many data sets to sort separately, and each one has only around 10 elements.

(e) You have a large data set, but all the data has only one of about 10 values for sorting purposes (e.g., the data is records of elementary-school students and the sort is by age in years).

(f) Instead of sorting the entire data set, you only need the k smallest elements where k is an input to the algorithm but is likely to be much smaller than the size of the entire data set.