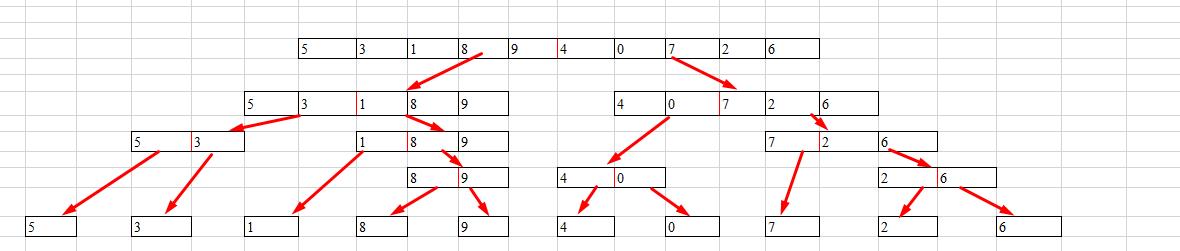
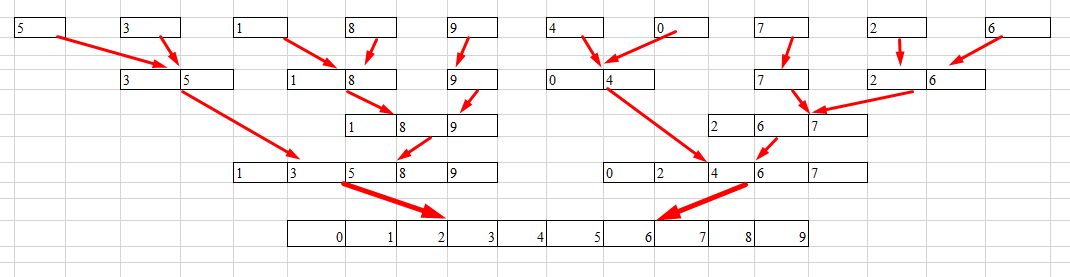
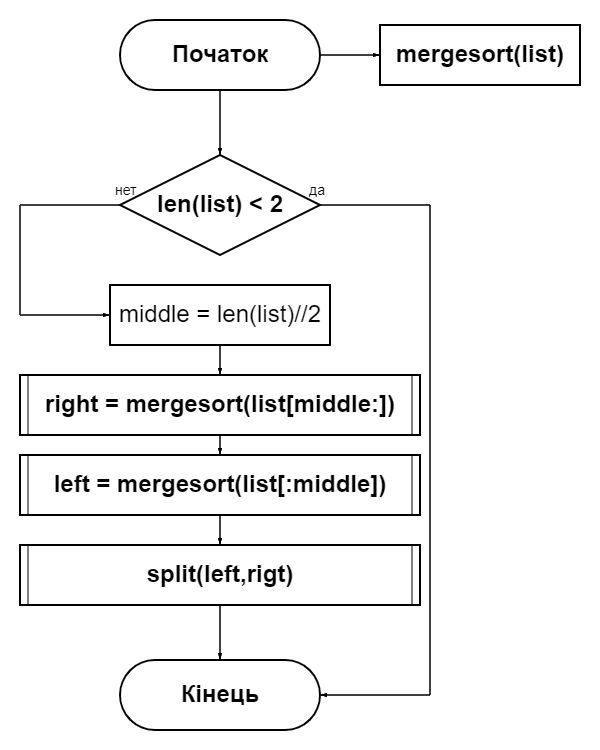
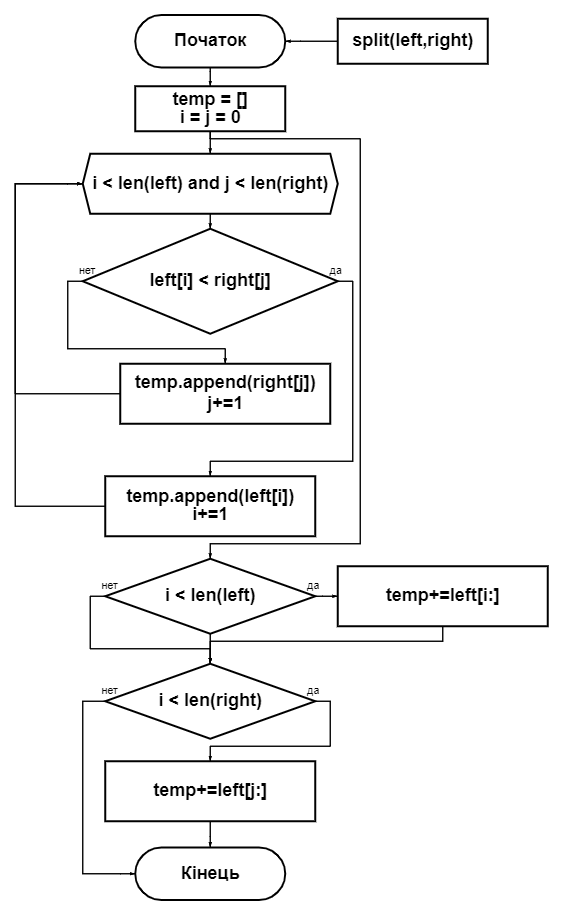
1. Own example of the corresponding sorting algorithm on an array of 10 numbers.

Breaking:

Merge:

Sorting algorithm in graphical form:

Block diagram of the "split" () function



In order to estimate the running time of a recursive algorithm, we must take into account the time spent on recursive calls. So, we get some recurrent relation, based on which we can estimate the running time of the algorithm.

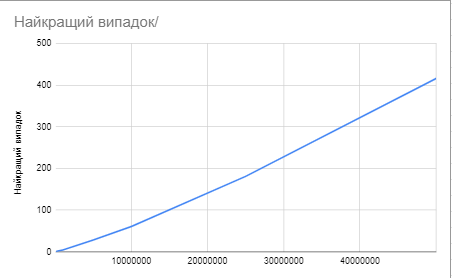
Where "n" - the size of the problem, "a" - the number of sub-tasks, "b" - a characteristic of how many times this sub-task is less than "n", "D (n)" - the time required for partitioning, "C (n) "- time required for connection.

It should be noted that this relationship holds for fairly large input sizes, when it makes sense to divide the subtask into subtasks. For small n, when partitioning is unnecessary or not possible, use the direct method of solving the problem.

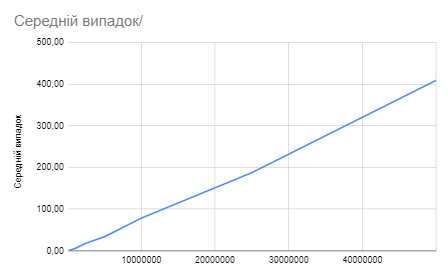
Analysis of the merge sorting algorithm. For simplicity, assume that the size of the array is a power of two. Then at each step the area to be sorted is divided into two equal halves. Breaking into parts (calculating the boundary) requires time θ (1) and merging - time θ (n).

So, solving this relationship we come to the following estimate:

best case schedule



schedule with the average case



worst case schedule

