**Exercise 1**

Suppose you are choosing between the following three algorithms.

* Algorithm A solves problems by dividing them into five subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.
* Algorithm B solves problems of size n by recursively solving two subproblems of size n – 1 and then combining the solutions in constant time.
* Algorithm C solves problems of size n by dividing them into nine subproblems of size n=3, recursively solving each subproblem, and then combining the solutions in O(n2) time.

What are the running times of each of these algorithms (in big-O notation), and which would you choose?

\* Do the exercise in your copybook (or on a separate sheet) and hand over to teacher for checking.

**Exercise 2**

Suppose you have *k* sorted arrays, each with *n* elements, and you want to combine them into a single sorted array of *kn* elements.

1. Using the **merge** procedure, merge the first two arrays, then merge in the third, then merge in the fourth, and so on. What is the time complexity of this algorithm, in terms of *k* and *n*?
2. Give a more efficient solution to this problem, using **divide-and-conquer**.

\* Implement a) and b) using any programming language. Elements ***n*** and ***k*** are entered from the keyboard by the user.

**Exercise 3**

Modify and implement the merge sort algorithm to sort an array of strings in lexicographic order.

**Exercise 4**

Implement procedure of Rearrange using any programming language (m and n are entered by the user).