**Lab 5**

**Mouna Kacimi, Werner Nutt, Camilo Thorne**

# Variants of Quicksort

As you may recall from the lectures, the main ideas behind QUICKSORT are two: **(1)** to recursively (via divide-and-conquer) partition the input array $A$ at **(2)** a partition point or *pivot* (via an auxiliary PARTITION algorithm). The algorithm discussed in the lecture chooses always a leftmost pivot. The aim of this lab is to see if alternative, correct versions of QUICKSORT can be obtained by considering alternative pivots and partition functions.

1. Define in pseudocode an alternative version of PARTITION where we choose as pivot the *median* element of the input array $A$.
   (a) How different is this new version from the algorithm seen in class?
   (b) Determine the loop invariant of this new version and prove its (partial) correctness.
   (c) Implement the algorithm in Java; i.e., implement a method `static public int partitionB(int[] A)`.
2. Define (also in in pseudocode) another alternative version of PARTITION where we choose as pivot the *rightmost* element of the input array $A$.
   (a) How different is this new version from the algorithm seen in class?
   (b) Determine the loop invariant of this second alternative version of partition and prove its (partial) correctness.
   (c) Implement the algorithm in Java; i.e., implement a method `static public int partitionC(int[] A)`.