For this lab you will practice with low-level reference-based lists. You will also use stacks as a substitute for recursion.

1. Write a method that receives integers \( n \) and \( m \) and builds and returns a list of length \( n \) containing nodes of type \( iNode \), as described in class, where every item is in the 0 to \( m - 1 \) range.

2. Write a method that sorts the lists of integers created by the previous method using mergesort. Adapt the standard mergesort algorithm to work with lists of elements of type \( iNode \).

3. Implement a non-recursive version of the method from the previous question. Modify your method to use a stack as a substitute for recursion. You may use the Java Stack class or implement your own. See handout for hints about how to do this.

Make your methods as efficient as possible; try to avoid traversing the list unnecessarily (for example to split the list or to find the number of nodes each partition should have). Run experiments with several different values of \( n \) and show your running times by means of plots and/or tables. Discuss whether the theoretical results match the outcomes of the experiments, making sure you evaluate the algorithms’ behavior with a wide range of values of \( n \). Write a report discussing your work, as described in the syllabus.