1. Use the following Python code to generate an input array:

   ```
   import numpy as np
   np.random.seed(1809xxx) # the number part of your student id
   np.random.choice(90, 10, replace=False) + 10
   ```

   and then sort the array by
   a. Mergesort (10 marks)
   b. Quicksort (10 marks)
   c. Heapsort (10 marks)
   d. Radix sort (10 marks)

   You need to show each major intermediate step of applying these sorting algorithms.

2. A positive integer array $A[1 \ldots n]$ is said to have a majority element if more than half of its entries are the same. Given an array, design an efficient algorithm with $O(nlgn)$ time to tell whether the array has a majority element, and, if so, to find that element. Otherwise, return 0. An algorithm with time complexity other than $O(nlgn)$ will receive no marks.
   a. Write pseudocode of your algorithm. (30 marks)
   b. Implement this algorithm in Python as a function:

      ```
      def find_majority_element(S):
      ```

      Two cases will be tested while one array has a majority element and one array does not have. The sum of running time is recorded. The fastest three implementations in the class will get a bonus 10 points for this assignment. (30 marks)