Project 4 Test sorting algorithms

1. In this project, implement insertion sort, selection sort, bubble sort, merge sort, quick sort implemented by user, together with sort function provided by C++ library, sort up to 1 million integers in **descending** order.
2. Normally we do complexity analysis for an algorithm by terms of problem size. The actual running time of an algorithm can be affected by algorithm itself, implementation and tuning details, and machine speed.
3. Requirements and hints
   (1) Randomly generate up to a million integers and put them in each array, one for each testing algorithm.
   (2) Since there are up to a million integers involved, we cannot manually verify that whether the result in actually in descending order. Need to implement a function isDescending to test whether the sorted array is actually in descending order.
   (3) Use clock function, which is included in library ctime, to get the current time in term of clock ticks before sorting, then call clock function again after sorting is finished, the difference between these two numbers, divided by CLOCKS_PER_SEC / 1000 is millisecond elapsed.
   (4) C++ has sort function, included in library algorithm, under namespace std. Compare your sorting algorithm with the one provided by C++ as well.
   (5) A sample output can be as follows. I did not enclose result of insertion sort, selection sort, and bubble sort on 1 million integers since it may take too much time (can be over one hour).

   quick sort: 156.019 ms
   merge sort: 301.648 ms
   system provided sort: 72.479 ms