Report of Exercise 2

1.Selection Sort

- The time complexity of Selection Sort is O(n^2). Because selection sort needs to find out the smallest item in the unsorted array, and put them in the front of the array, finding the element need to take O(n) times. This kind of implementations need to be repeated as n times, so the time complexity is O(n^2).
- The average sort time for 3 runs is 3540927463800, the standard deviation is 228019472646.

2.Insertion Sort

- The time complexity of Insertion Sort is O(n^2). Because insertion sort will get an element as the fundamental element, and search in the rest of the unsorted array, put the smaller element in the front of the fundamental element, and put the larger element in the back of the fundamental element, in this way the array needs to be scaned n times. As there is n elements, the scan time would be n*n, the time complexity of Insertion Sort is O(n^2).
- The average sort time for 3 runs is 878423967300, the standard deviation is 3069827612.

3.Heap Sort

- The time complexity of Heap Sort is O(nlogn). Because we need to heapify a max heap and than find the biggest element of this heap. The finding time of each max element will take logn times, and we need to do this implemention for like n times. So the time complexity of heap sort is O(nlogn).
- The average sort time for 10 runs is 1546454200, the standard deviation is 136694113.

4.Merge Sort

- The worst time complexity of Merge Sort is O(nlogn). Because the array needs to be split into two parts, and four quarter, … until each part has only one element. This operation needs to be done logn times. And in each small part, the elements will be compared and reordered, so they will have the right order in the part. The comparision needs to be done n times, so the complexity of merge sort is O(nlogn).
- The average sort time for 10 runs is 3362378200, the standard deviation is 129133246.

5.Quick Sort

- The worst time complexity of Quick Sort is O(n^2), but the average time complexity of Quick Sort is O(nlogn). Firstly we choose an element as pivot. Then find out all of the elements which is smaller than pivot, put them in the front of the pivot, and find out all of the elements which is larger than pivot, put them in the back of the pivot. Choose another pivot and do the same operation for each pivot, recursively we can reorder the whole array. Each of the comparision for this sorting is n times, and this kind of comparision needs to be executed for logn times. So the average time complexity of this algorithm is O(nlogn). However, if we always choose the smallest element of the rest array, we will get a time complexity of O(n^2)
- The average sort time for 10 runs is 10981166900, the standard deviation is 1452497941.

	Mean Time	Standard Deviation
Selection Sort	59mins	3.8mins
Insertion Sort	14.6mins	3secs

	Mean Time	Standard Deviation
Heap Sort	1.5secs	136msecs
Merge Sort	3secs	129msecs
Quick Sort	10secs	1sec