【DS】Day10

Tags	
Date	@June 3, 2022
Summary	Quicksort and Quicksele

【Week 3】Quicksort

3.4 Quicksort

- Shuffle the array
- Partition so that, for some j
    - entry a[j] is in place
    - no large entry to the left of j
    - no smaller entry to the right of j
- Sort each piece recursively

```
private static int partition(Comparable[] arr, int lo, int hi ) {
  int i = lo, j = hi + 1;

  while (true) {
    while (less(arr[++i], arr[lo])
      if (i == hi) break;

    while (less(arr[lo], arr[--j])
      if (j == lo) break;

    if (i >= j) break;
    exch(arr, i, j);
  }
  exch(arr, lo, j);
  return j; // Return index of item now known to be in place
}
```

```
public class Quick {
  private static int partition(Comparable[] ar, int lo, int hi) { ... }
  public static void sort(Comparable[] arr) {
    StdRandom.shuffle(arr);
    Quick.sort(arr, 0, arr.length - 1);
  }

  private static void sort(Comparable[] arr, int lo, int hi) {
    if (hi <= lo) return;
    int j = partition(arr, lo, hi);
    sort(arr, lo, j - 1);
    sort(arr, j + 1, hi);
  }
}
```

3.5 Selection

Goal: Given an array of N items, find the kth largest.

Repeat partition in one subarray, depending on j; finished when j equals k

```
public static Comparable select(Comparable[] arr, int k) {
  StdRandom.Shuffule(arr);
  int lo = 0, hi = arr.length - 1;
  while (hi > lo) {
    int j = partition(arr, lo, hi);
    if (j > k) hi = j - 1;
    else if (j < k) lo = j + 1;
    else return a[k];
  }
  return a[k];
}
```