Quick Sort

3111 손완서 4/8 (목) 발표

이론

- Quick Sort는 기준점을 획득한 다음 해당 기준점의 좌우를 정렬한 후 배열을 나눈다.
- 이런 식으로 모든 항목이 정렬될 때까지 이 과정을 반복한다.
- 가장 이상적인 기준점은 배열의 중간 값(배열을 균등하게 나눌 수 있기 때문)
  하지만 배열의 중간 값을 얻으려면 선형 시간이 걸린다.

코드 구현

```
function quickSort(items) {
  return quickSortHelper(items, 0, items.length - 1);
}

function quickSortHelper(items, left, right) {
  let index;

  if(items.length > 1) {
    index = partition(items, left, right);

    if(left < index - 1) quickSortHelper(items, left, index - 1);

    if(index < right) quickSortHelper(items, index, right);
  }

  return items;
}

function partition(array, left, right) {
  let pivot = array[Math.floor((right + left) / 2)];

  while(left <= right) {
    while(pivot > array[left]) left++;
        while(pivot < array[right]) right--;

    if(left <= right) {
      var temp = array[left];
      array[left] = array[right];
      array[right] = temp;
      left++;
      right--;
    }
  }

  return left;
}

quickSort([6, 1, 23, 4, 2, 3]); // [1, 2, 3, 4, 6, 23]
```

```
[ 6, 1, 23, 4, 2, 3 ] 0 5
[ 6, 1, 3, 4, 2, 23 ] 0 4
[ 2, 1, 3, 4, 6, 23 ] 0 2
[ 1, 2, 3, 4, 6, 23 ] 1 2
[ 1, 2, 3, 4, 6, 23 ] 3 4
```

질문

- Quick Sort보다 빠른 정렬이 있는가?
  - 계수 정렬: O(n)

- Quick Sort 개념, 특징, 시간 복잡도, 공간 복잡도
  - pivot을 기준으로 좌우를 정렬한 후 배열을 다시 분해
  - 속도가 빠름, 정렬된 리스트에 대해서는 오히려 수행시간이 더 많이 걸린다.
  - 평균: O(n log 2(n)), 최악: O(n^2)
  - 재귀 함수의 콜 스택 때문에 O(log 2 (n))

- 어떨 때 사용할까?
  - 평균 성능이 최적화되어야 하는 상황에 사용