JavaScript 정렬 알고리즘
1) 선택 정렬

선택 정렬 | 알고리즘의 기본이 되는 정렬 알고리즘 이해하기

강사 나동빈

JavaScript 정렬 알고리즘

1) 선택 정렬

선택 정렬(Selection Sort)

- 선택 정렬은 **매 단계**에서 가장 작은 원소를 **선택**해서 앞으로 보내는 정렬 방법이다.
- 앞으로 보내진 원소는 더 이상 위치가 변경되지 않는다.
- 시간 복잡도 $O(N^2)$로 비효율적인 정렬 알고리즘 중 하나다.

선택 정렬(Selection Sort) 동작 방식

1. 각 단계에서 가장 작은 원소를 **선택**한다.
2. 현재까지 처리되지 않은 원소들 중 **가장 앞의 원소와 위치를 교체**한다.

선택 정렬(Selection Sort) 예시

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

선택 정렬(Selection Sort) 예시

JavaScript 정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[1단계]

1	4	3	2	9	6	8	7	5

선택 정렬(Selection Sort) 예시

JavaScript 정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[1단계]

1	4	3	2	9	6	8	7	5

[2단계]

1	2	3	4	9	6	8	7	5

선택 정렬(Selection Sort) 예시

JavaScript 정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[1단계]

1	4	3	2	9	6	8	7	5

[2단계]

1	2	3	4	9	6	8	7	5

[3단계]

1	2	3	4	9	6	8	7	5

선택 정렬(Selection Sort) 예시

JavaScript 정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[1단계]

1	4	3	2	9	6	8	7	5

[2단계]

1	2	3	4	9	6	8	7	5

[3단계]

1	2	3	4	9	6	8	7	5

[4단계]

1	2	3	4	9	6	8	7	5

선택 정렬(Selection Sort) 예시

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[1단계]

1	4	3	2	9	6	8	7	5

[2단계]

1	2	3	4	9	6	8	7	5

[3단계]

1	2	3	4	9	6	8	7	5

[4단계]

1	2	3	4	9	6	8	7	5

[5단계]

1	2	3	4	5	6	8	7	9

선택 정렬(Selection Sort) 예시

JavaScript 정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[6단계]

1	2	3	4	5	6	8	7	9

선택 정렬(Selection Sort) 예시

JavaScript 정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[6단계]

1	2	3	4	5	6	8	7	9

[7단계]

1	2	3	4	5	6	7	8	9

선택 정렬(Selection Sort) 예시

JavaScript 정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[6단계]

1	2	3	4	5	6	8	7	9

[7단계]

1	2	3	4	5	6	7	8	9

[8단계]

1	2	3	4	5	6	7	8	9

선택 정렬(Selection Sort) 예시

JavaScript
정렬
선택 정렬

- 정렬할 배열:

2	4	3	1	9	6	8	7	5

: 정렬 완료

[6단계]

1	2	3	4	5	6	8	7	9

[7단계]

1	2	3	4	5	6	7	8	9

[8단계]

1	2	3	4	5	6	7	8	9

[정렬 완료]

1	2	3	4	5	6	7	8	9

선택 정렬(Selection Sort) 소스 코드 예시


```
// 선택 정렬 함수
function selectionSort(arr) {
  for (let i = 0; i < arr.length; i++) {
    let minIndex = i; // 가장 작은 원소의 인덱스
    for (let j = i + 1; j < arr.length; j++) {
      if (arr[minIndex] > arr[j]) {
        minIndex = j;
      }
    }
    // 스와프(swap)
    let temp = arr[i];
    arr[i] = arr[minIndex];
    arr[minIndex] = temp;
  }
}
```

선택 정렬(Selection Sort) 소스 코드 예시


```
/* 1) 선택 정렬의 수행 시간 측정 */
// 0부터 999까지의 정수 30000개를 담은 배열 생성
let arr = Array.from({ length: 30000 }, () => Math.floor(Math.random() * 1000));

// getTime(): 1970-01-01부터의 시간차를 ms 단위로 계산
let startTime = new Date().getTime();
selectionSort(arr);
let endTime = new Date().getTime();

// 시간차 출력
console.log('선택 정렬 소요 시간:', endTime - startTime, "ms.");

/* 2) 이미 정렬된 배열에 대한 선택 정렬의 수행 시간 측정 */
// 모든 값이 7인 정수 30000개를 담은 배열 생성
arr = Array.from({ length: 30000 }, () => 7);

// getTime(): 1970-01-01부터의 시간차를 ms 단위로 계산
startTime = new Date().getTime();
selectionSort(arr);
endTime = new Date().getTime();

// 시간차 출력
console.log('정렬된 배열에 대한 선택 정렬 소요 시간:', endTime - startTime, "ms.");
```

[실행 결과]

```
선택 정렬 소요 시간: 1743 ms.
정렬된 배열에 대한 선택 정렬 소요 시간: 1895 ms.
```

선택 정렬(Selection Sort)의 시간 복잡도

- **선택 정렬**이란 가장 작은 것을 선택해서 앞으로 보내는 정렬 기법이다.
- 매 단계에서 가장 작은 것을 선택하는 데에 약 N번의 연산이 필요하다. (선형 탐색)
- 결과적으로 약 N개의 단계를 거친다는 점에서 최악의 경우 $O(N^2)$의 시간 복잡도를 가진다.