배열과객체

배열과 객체: CRUD 개념을 중심으로

1. 배열과 객체의 공통점: CRUD 기능

배열과 객체는 모두 데이터를 저장하고 조작할 수 있는 자료구조입니다. 이 둘의 가장 중요한 공통점 중 하나는 **CRUD** 기능을 지원한다는 점입니다.

CRUD는 다음과 같습니다:

- **Create**: 데이터를 추가한다.
- **Read**: 데이터를 읽는다.
- **Update**: 데이터를 수정한다.
- **Delete**: 데이터를 삭제한다.

배열과 객체는 모두 이러한 기본적인 동작을 지원하지만, 데이터에 접근하고 관리하는 방식에서 차이가 있습니다.

2. 배열: 인덱스를 통한 데이터 접근

배열은 **인덱스**라는 번호를 통해 데이터를 관리하고 접근합니다. 배열은 순서가 있는 데이터의 모음이며, 데이터는 0부터 시작하는 인덱스로 접근할 수 있습니다.

배열 예시:

```
// 배열 생성 (Create)
const fruits = ['사과', '바나나', '오렌지'];

// 배열 요소 읽기 (Read)
console.log(fruits[0]); // '사과'

// 배열 요소 수정 (Update)
fruits[1] = '딸기';
console.log(fruits); // ['사과', '딸기', '오렌지']
```

```
// 배열 요소 삭제 (Delete)
fruits.splice(2, 1); // 세 번째 요소인 '오렌지' 삭제
console.log(fruits); // ['사과', '딸기']
```

3. 객체: 키와 값으로 데이터 접근

객체는 **키(key)**와 **값(value)**의 쌍으로 데이터를 저장하고 접근합니다. 각 키는 고유하며, 해당 키를 통해 값에 접근하거나 수정할 수 있습니다.

객체 예시:

```
// 객체 생성 (Create)
const person = {
  name: '홍길동',
  age: 25,
  city: '서울'
};

// 객체 속성 읽기 (Read)
console.log(person.name); // '홍길동'

// 객체 속성 수정 (Update)
person.age = 26;
console.log(person); // { name: '홍길동', age: 26, city: '서
울' }

// 객체 속성 삭제 (Delete)
delete person.city;
console.log(person); // { name: '홍길동', age: 26 }
```

4. 배열과 객체의 차이점

- **데이터 접근 방식**: 배열은 인덱스를 통해 순서대로 데이터를 접근하고, 객체는 키(key)를 통해 값을 접근합니다.
- **순서**: 배열은 데이터에 순서가 있으며, 객체는 순서와 상관없이 키를 통해 값을 찾습니다.

5. 배열과 객체의 유사성

배열과 객체는 둘 다 **CRUD** 기능을 제공합니다. 즉, 데이터를 생성하고 읽으며, 수정하고 삭제하는 기본적인 기능을 동일하게 지원합니다. 배열은 객체의 일종이기도 하기 때문에, 배열 안에서도 객체와 같은 CRUD 작업을 수행할 수 있습니다.
