2주차 4차시

11장 과제

1. **밑의 코드를 실행해보고 이유를 생각해보시오.**

```
const original = [
[18, 18, 18, 18],
[19, 19, 19, 19],
[20, 20, 20, 20],
[21, 21, 21, 21],
];
const copy = original.slice();
console.log(JSON.stringify(original) === JSON.stringify(copy));
copy[0][0] = 99;
copy[2].push("02");
console.log(JSON.stringify(original) === JSON.stringify(copy));
console.log(original);
console.log(copy);
```

```
true
true
(4) [Array(4), Array(4), Array(5), Array(4)]
(4) [Array(4), Array(4), Array(5), Array(4)]
```

2. **밑에 있는 코드들을 실행해 보시고 이유를 생각해보세요.**

```
const obj = { a: 1 };
const newObj = Object.assign({}, obj);
newObj.a = 2;
console.log(obj);
console.log(obj === newObj);
```

```
{a: 1}
false
```

```
const obj = {
a: 1,
b: {
c: 2,
},
};
```

```
const newObj = Object.assign({}, obj);
newObj.b.c = 3;
console.log(obj);
console.log(obj.b.c === newObj.b.c);
```

```
{a: 1, b: {…}}
true
```

3. 알잘딱 위처럼!

```
const obj = { a: 1 };
const newObj = Object.assign({}, obj);
newObj.a = 2;
console.log(obj);
console.log(obj === newObj);
```

```
{a: 1}
false
```

```
const obj = {
a: 1,
b: {
c: 2,
},
};
const newObj = { ...obj };
newObj.b.c = 3;
console.log(obj);
console.log(obj.b.c === newObj.b.c);
```

```
{a: 1, b: {…}}
true
```

4. 함수를 작동시켜 보세요

```
function deepCopy(obj) {
if (obj === null || typeof obj !== "object") {
return obj;
}
let copy = {};
for (let key in obj) {
copy[key] = deepCopy(obj[key]);
```

```
}
return copy;
}
const obj = {
a: 1,
b: {
c: 2,
},
func: function () {
return this.a;
},
};
const newObj = deepCopy(obj);
newObj.b.c = 3;
console.log(obj);
console.log(obj.b.c === newObj.b.c);
```

```
{a: 1, b: {…}, func: ƒ}
false
```

5. **Lodash 라이브러리에 대해서 조사하고 cloneDeep 메서드도 조사해보세요.**
    a. Lodash는 JavaScript의 라이브러리 중 하나로 array, collection, date 등 데이터의 필수적인 구조를 쉽게 다룰 수 있게 도와준다. 특히 배열 안의 객체들의 값을 핸들링 할 때 유용하다.
    b. cloneDeep 메서드는 객체 타입을 복사하는 메소드이다. 깊은 복사(Deep Clone)은 복사된 모든 값이 내부 자식 요소를 포함하여 모두 참조 형태가 아닌 새로운 값이 매핑되는 형태로 복사되는 것을 말하며 이를 위해 cloneDeep 메소드가 사용된다.
6. **소개한 방법 이외에도 깊은복사, 얕은복사 방법을 찾아보세요**
   가능하다면 언어 별로 정리해도 좋고, JS만 하셔도 좋고, 단일 언어 하나만 하셔도 좋습니다.
    a. **C**

```
// 얕은 복사
typedef struct name
{
  char* lastName;
  char* firstName;
} name_t;

void main()
{
  char firstName[] = "Lulu";
  char lastName[] = "Lee";
```

```
  name_t name;
  name_t clone;

  name.lastName = lastName;
  name.firstName = firstName;

  clone = name;
  name.lastName[0] = 'N';

  printf("original: %s %s\n", name.firstName, name.lastName);
  printf("clone: %s %s\n", clone.firstName, name.lastName);
}
```

```
// 깊은 복사
typedef struct name
{
  char* lastName[64];
  char* firstName[64];
} name_t;

void main()
{
  name_t name = {"Lulu", "Lee"};
  name_t clone;

  clone = name;
  name.lastName[0] = 'N';

  printf("original: %s %s\n", name.firstName, name.lastName);
  printf("clone: %s %s\n", clone.firstName, name.lastName);
}
```

b. JavaScript

```
// 얕은 복사
const original = ['a',2,true,4,"hi"];
const copy = original.slice();
```

```
// 깊은 복사
var original = {
  a: 10,
  b: 'abc',
};
var copy = JSON.parse(JSON.stringify(original));
copy.b = 3;
console.log(original);
console.log(copy);
```