

**Question 1]** function goNext() 의 qnaList[qIdx].a[i].answer 가 의미하는 것이 무엇인지 설명하시오.

```
function goNext(qIdx){
    var q = document.querySelector('.qBox');
    q.innerHTML = qnaList[qIdx].q;
    for(let i in qnaList[qIdx].a){
        addAnswer(qnaList[qIdx].a[i].answer, qIdx, i); }}}
```

**Question 2]** function addAnswer() 의 아래 두 붉은 박스에 대하여 설명하시오.

```
function addAnswer(answerText, qIdx, idx){
    var a = document.querySelector('.answerBox');
    var answer = document.createElement('button');
    answer.classList.add('answerList', 'my-3', 'py-3', 'mx-auto', 'fadeIn');

    a.appendChild(answer);
    answer.innerHTML = answerText;

    answer.addEventListener("click", function(){
        var children = document.querySelectorAll('.answerList');
        for(let i = 0; i < children.length; i++){
            children[i].disabled = true;

            children[i].style.WebkitAnimation = "fadeOut 0.5s";
            children[i].style.animation = "fadeOut 0.5s";
        }
        setTimeout(() => {
            var target = qnaList[qIdx].a[idx].type;
            for(let i = 0; i < target.length; i++){
                select[target[i]] += 1;
            }
            for(let i = 0; i < children.length; i++){
                children[i].style.display = 'none';
            }
            goNext(++qIdx);
        }, 450)
    }, false);
}
```

**Question 3]** function calResult() 의 {} 안의 각 줄마다 어떠한 기능인지 설명하시오.

```
function calResult(){  
  console.log(select);  
  var result = select.indexOf(Math.max(...select));  
  return result;}  
}
```

**Question 4]** function goNext() 에서 기존에 학습했던 i.length 가 아닌 in qnaList 를 쓴 이유를 설명하시오.

```
function goNext(qIdx){  
  var q = document.querySelector('.qBox');  
  q.innerHTML = qnaList[qIdx].q;  
  for(let i in qnaList[qIdx].a){  
    addAnswer(qnaList[qIdx].a[i].answer, qIdx, i); }}  
}
```

Answer 1] : 배열안의 객체와 해당 객체의 값을 가지는 배열로 이루어진 객체를 의미

- `qnaList[qIdx] : qnaList = [ ]`
  - `[ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
- `qnaList[qIdx].q : qnaList = [ { q, a } ]`
  - `[ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
- `qnaList[qIdx].a : qnaList = [ { q, a } ]`
  - `[ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
- `qnaList[qIdx].a[i] : qnaList = [ { q, a [i] } ]`
  - `a[0] : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
  - `a[1] : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
  - `a[2] : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
- `qnaList[qIdx].a[i].answer : qnaList = [ { q, a[i].answer } ]`
  - `a[0].answer : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
  - `a[1].answer : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
  - `a[2].answer : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`

```
const qnaList = [
  {
    q: '1. 이성 사이에 진정한 친구는 있다, 없다?',
    a: [
      { answer: 'a. 이성 사이에 친구가 어딴? 절대 없어', type: [1, 2, 4, 9] },
      { answer: 'b. 친구 있지, 절대 이성으로만 안 보일뿐', type: [0, 3, 6, 5, 10, 8] },
      { answer: 'c. 난 잘 모르겠어..', type: [7, 11] },
    ]
  },
];
```

*data.js*

- 객체의 간단한 예시는 아래와 같으며, `name`: 와 `q`: 가 각각의 자료값을 가지고 있습니다.

```
let person = {name:"강원일", age:25};
```

*age* 의 *property* 가 이상하다?!

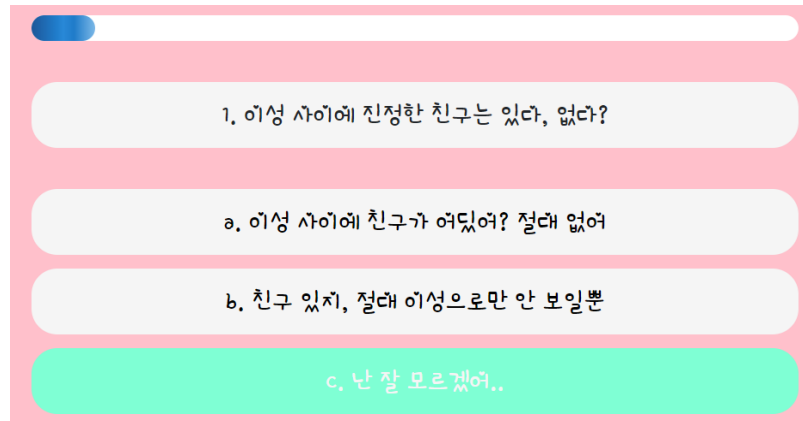
- 자바스크립트에서의 자료의 나열을 `data.js` 처럼 배열과 객체의 복합적인 구조를 가지기도 합니다.
- 위와 같은 `data.js` 에서 원하는 값을 뽑아오지 못한다면,  
즉, `qnaList[qIdx].a[i].answer` 가 어느 위치를 의미하는지 잘 모른다면 중요한 학습지점을 놓치는 것 입니다.
- `function addAnswer()` 의 `Array` 와 `object` 도 유사합니다.

```
var target = qnaList[qIdx].a[idx].type;
for(let i=0; i<target.length; i++){
  select[target[i]] += 1
}
```

- `qnaList[qIdx].a[idx].type : qnaList = [ { q, a[i].type } ]`
  - `a[0].type : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
  - `a[0].type : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`
  - `a[0].type : [ { q, a [ { answer, type[] }, { answer, type[] }, { answer, type[] } ] } ]`

Answer 2] : 각각 답변을 클릭시 추가 클릭 불가, 화면에서 사라짐을 의미

- children[i].disabled = true;
  - C를 선택함과 동시에 다른 선택지를 선택하지 못하도록 한다.



하나의 선택지만을 강제

- children[i].style.display = 'none';
  - 선택과 동시에 해당 선택지는 사라지게 함

```
function addAnswer(answerText, qIdx, idx){
  var a = document.querySelector('.answerBox');
  var answer = document.createElement('button');
  answer.classList.add('answerList', 'my-3', 'py-3', 'mx-auto', 'fadeIn');

  a.appendChild(answer);
  answer.innerHTML = answerText;

  answer.addEventListener("click", function(){
    var children = document.querySelectorAll('.answerList');
    for(let i = 0; i < children.length; i++){
      children[i].disabled = true;

      children[i].style.WebkitAnimation = "fadeOut 0.5s";
      children[i].style.animation = "fadeOut 0.5s";
    }
    setTimeout(() => {
      var target = qnaList[qIdx].a[idx].type;
      for(let i = 0; i < target.length; i++){
        select[target[i]] += 1;
      }
      for(let i = 0; i < children.length; i++){
        children[i].style.display = 'none';
      }
      goNext(++qIdx);
    }, 450);
  }, false);
}
```

- button의 click은 answerList(a. b. c. 난 잘모르겠어) 중 하나가 선택된 것
- 선택과 동시에 disabled(추가 선택 불가)과 선택지가 사라짐(display=none)

### Answer 3] : select 의 최빈값을 도출

```
function calResult(){
  console.log(select);
  var result = select.indexOf(Math.max(...select));
  return result;}

```

강의중 최빈값을 도출한다고만 언급

- ...select : [0,0,0,0,0,0,0,0,0,0,0] 를 의미
- Math.max() : ( ) 안의 배열중 최대값
- select.indexOf(n) : select 배열중 (n, 조건) 이 몇번째에 있는가
- select.indexOf(Math.max()) : select 배열중 최대값이 몇번째에 있는가
  - 본문 4page 의 <추가 사항> 참고
- return result : 해당 결과값을 출력(반환), 만약 result 가 6번째면 calResult() 는 6

<보충 사항>

- indexOf : 문자열에서 원하는 문자의 위치를 찾거나 배열에서는 배열 값의 존재 여부를 확인하는 방법

문자열: hello world										
h	e	l	l	o	공백	w	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10

```
let str = 'hello world';
str.indexOf('e'); // 1
str.indexOf('E'); // -1
str.indexOf('a'); // -1

```

- 문자열에서 'e'가 존재하는 index는 1이므로 1을 반환
- 대문자 'E'와 'a'는 문자열에 존재하지 않으므로 -1을 반환

```
let str = 'hello world';
str.indexOf('o', 4); // 4
str.indexOf('o', 5); // 7
str.indexOf('o', -10); // 4

```

- fromIndex값이 4이므로 str의 index 4(o world)부터 'o'를 탐색한다.
- 탐색한 'o'의 실제 index인 4를 반환한다.
- 마찬가지로 fromIndex 5는 (공백) world부터 탐색하여 hello에 있는 'o'는 무시하고 world에 있는 'o'를 탐색하여 7을 반환한다.
- fromIndex값이 음수인 경우는 기본값인 0을 기준으로 문자열 전체를 탐색, 4 반환

- ... : 배열 앞쪽에 위치한 값 몇개만 필요, 그 이후 이어지는 나머지 값(rest)들을 모아서 저장

```
let [name1, name2, ...rest] = ["Julius", "Caesar", "Consul", "of the Roman Republic"];
console.log(name1);    Julius
console.log(name2);    Caesar
console.log(rest[0]);   Consul
console.log(rest[1]);   of the Roman Republic
console.log(rest.length); 2

```

## <추가 사항>

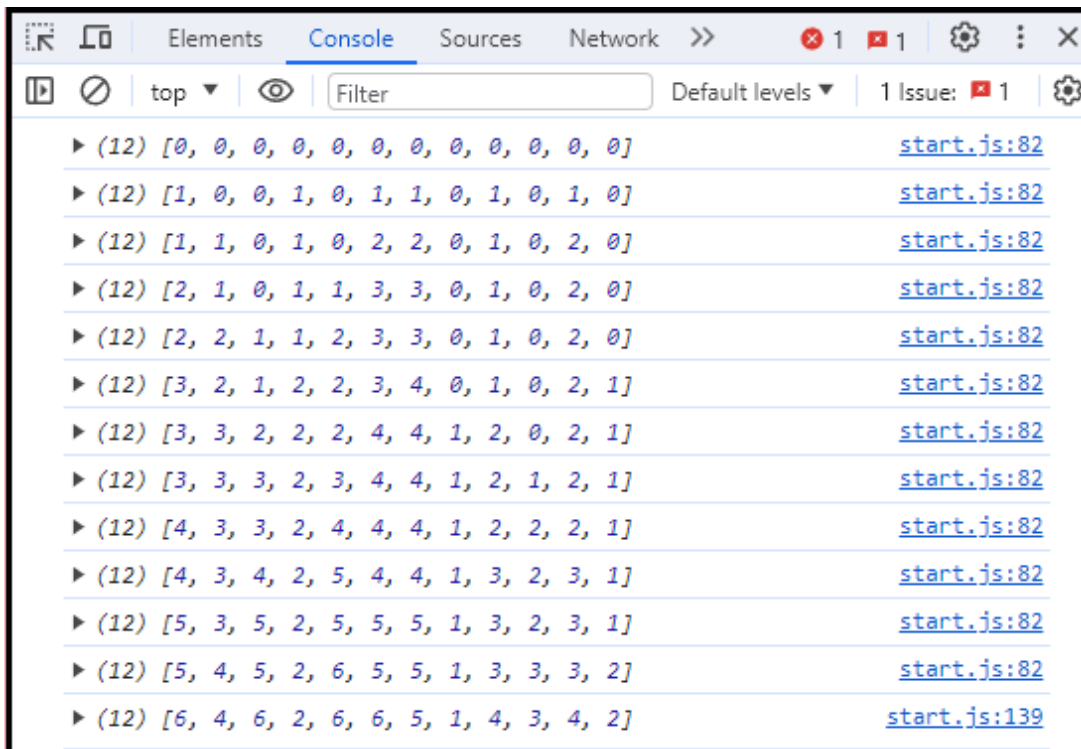
- 초기 배열의 상태는 아래와 같습니다.

```
let select = [0,0,0,0,0,0,0,0,0,0,0,0];
```

- 유저가 선택지들을 선택하면 type 의 값이 반영되어 해당하는 배열의 순서의 값이 증가합니다.

```
const qnaList = [
  {
    q: '1. 이성 사이에 진정한 친구는 있다, 없다?',
    a: [
      { answer: 'a. 이성 사이에 친구가 어딴어? 절대 없어', type: [1, 2, 4, 9] },
      { answer: 'b. 친구 있지, 절대 이성으로만 안 보일뿐', type: [0, 3, 6, 5, 10, 8] },
      { answer: 'c. 난 잘 모르겠어..', type: [7, 11] },
    ]
  }
],
```

- 1번 질문에 a를 선택하면 type [1,2,4,9]가 적용됨
  - [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] 에서
  - [0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0] 로 해당하는 수치가 1씩 증가
- 전체 과정을 개발자도구(f12)를 통해 확인하면 아래와 같이 수치가 변동(증가)합니다.



- 마지막 배열의 [6,4,6,2,6,6,5,1,4,3,4,2] 는 순서대로 쥐, 소, 호랑이, 토끼 등등의 순서입니다. 앞선 calResult의 설명처럼 이중 최고로 높은 값(Math.max())을 찾아서 반영한것이 위의 이미지입니다.
- 마지막 [6,4,6,2 ...] 처럼 결과값이 여러개일경우 세부적인 if 문이 필요한 편입니다.
- 간략한 학습을 위해 if문을 상세히 작성하지 않고, 배열의 순서상 좌측값을 우선 선택된 경우입니다.
  - 즉, 쥐>소>호랑이 순서로 좌측의 값이 더 많이 선택될 가능성이 높습니다.

Answer 4] : Array 와 Object 의 차이점에 관한 질문입니다. object 는 for in 을 통해 모든 값을 가져올 수 있습니다.

- 배열(Array) 와 객체(Object) 의 차이점
  - Array : 값들에 순서를 부여한다. [  
■ 책꽂이와 유사하게 책을 담을 때 순서는 중요하지 않음, 도서관리와 유사
  - 순회(iteration) : 순서대로 모든 요소에 차례로 접근할 수 있는 것
  - Object : 순서없이 값들을 저장한다. { }  
■ 값들은 각각의 이름을 가지며 값들의 이름을 통해 읽고 쓸 수 있다.
  - 객체는 정해진 순서는 없으나, for in 을 통해 모든 값을 가져올 수 있다.

```
for(let i in qnaList[qIdx].a)
```

- 우리는 Array 에 관해서는 상세히 진행했으나 object는 비교적 깊게 살펴본 바 없습니다.

<추가 사항>

- 아래의 Reference 는 단순 구글링을 했으며, 개인적으로 요약한 내용은 아래와 같습니다.  
[자바스크립트 - 객체 프로퍼티\(property\) 파헤치기!](#)  
[\[Javascript\] 객체의 프로퍼티에 접근하는 방법](#)  
[Chapter 13. 객체의 이해](#)

- 객체(object) 정의<sup>1</sup>
  - 물리적으로 존재하거나 추상적으로 생각할 수 있는 것 중에서 자신의 속성을 갖고, 다른 것들과 식별이 가능한 것을 의미
  - 다양한 변수와 함수중 연관된 것끼리 묶어 정리하는 개념

```
let person = {name:"김민현", age:25};
console.log(person);           { name: '김민현', age: 25 }
console.log(person.name);      김민현
console.log(person.age);       25
```

- 객체 person 은 name 이라는 변수에 '김민현'이라는 데이터를 담고 있다.
- name 과 age 라는 공통점으로 연관된 것들로 구성할 수 있다.

- 자바스크립트에는 8가지 자료형이 존재한다. 원시와 참조타입으로 구별된다.
  - 원시타입(primitive type) : 하나의 값을 가질 수 있다.
    - num, string, int, boolean, null, undefined, symbol
  - 참조타입(reference type) : 여러 값들을 포함할 수 있는 하나의 덩어리
    - Object

---

<sup>1</sup> [자바스크립트 - 객체 프로퍼티\(property\) 파헤치기!](#)  
[\[Javascript\] 객체의 프로퍼티에 접근하는 방법](#)  
[Chapter 13. 객체의 이해](#)

- 구조

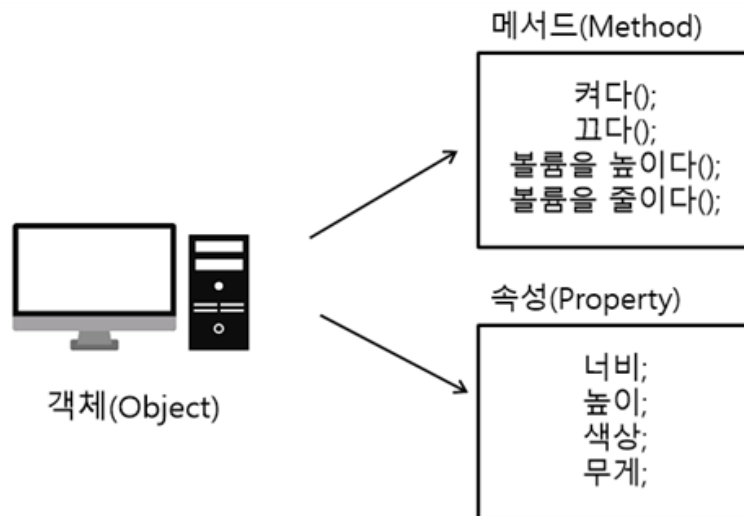
- `let a = {}`, 자료의 주소값을 집합 단위로 기억하며 {중괄호}로 작성함
- `Object.Method(Parameter)`

```
let kc = document.getElementsByName("city");
```

#### Dot Notation

- `document` : 객체
- `getElementsByName` : Method, 객체에 속하는, 포함되는 함수
- `"city"` : 파라미터

- `property` : 객체 내부의 속성, 객체에 포함된 변수들



- 프로퍼티의 이름과 값을 표기할 때 가급적 '알파벳으로 이루어진 문자열로 작성할 것'

- 객체에 접근하는 2가지 방법

- Dot Notation : 마침표(.)으로 접근
- Bracket Notation : 대괄호 [ ] 사이에 키값을 문자열로 넣어 접근

- Dot Notation 의 예시

```
let person1 = {}

person1.name = "김민현";
person1.age = 17;

console.log(person1);
console.log(person1.name);
console.log(person1.age);
```

{ name: '김민현', age: 17 }  
김민현  
17

빈 객체에 삽입

- `object` : `person` ex) 상품 목록, 학생의 이름과 번호
- `elements` : `name:"김민현", age:25`
  - 이때의 `element` 를 `key` 라고 하며 하나의 쌍을 의미한다.
- 프로퍼티 식별자는 오로지 알파벳만 가능하다.
- 숫자로 시작할 수 없으며, 변수를 포함할 수 없다



- Bracket Notation 의 예시

```
let myself={
  name : 'Code Kim',
  age : 30,
  location : {
    country : 'South Korea',
    city : 'Seoul'
  }
}

console.log(myself['name']);      Code Kim
console.log(myself['age']);       30
console.log(myself['location']);  { country: 'South Korea', city: 'Seoul' }
```

- key 값을 문자열로 넣어 접근하는 것을 명심하자
- 속성(property) 자체는 또다른 key 값을 가질 수 있다.
- 브라켓 표기법은 점표기법과 달리 프로퍼티에 대한 제약이 없다.
- 숫자로 시작할 수 있으며 변수, 공백 사용이 가능

- JSON(JavaScript Object Notation) : 자바스크립트에서 사용하는 객체의 규격

```
function goNext(qIdx){
  if(qIdx === endPoint){
    let selectValue = {};
    selectValue.select = select;
    sessionStorage.setItem("selectValue", JSON.stringify(selectValue));
    console.log(selectValue);
    location.href = "result.html"
    return;
  }
}
```

11.20 추가됨