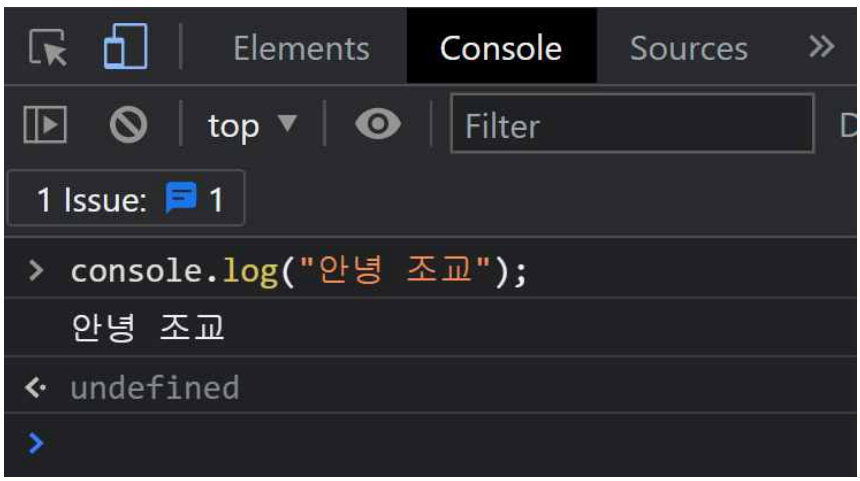
1장 변수 설명 전까진
JavaScript가 어떤 언어인지 쭉 설명할 것이다.
어려운 개념을 엄청나게 설명할 것이지만
그냥 마음 편하게 듣고 1장 변수설명할때부터 집중해서 듣길 바람

HTML은 마크업 언어였고,
CSS는 스타일시트 언어였다.
즉, HTML과 CSS까지 알고있다면 개발자가 아니라
웹 퍼블리셔, 또는 웹 디자이너의 기초를 배운 것이다.
JavaScript는 브라우저에서 쓰이는 단 하나의 "프로그래밍 언어"다.
W3C 에서 다른 프로그래밍 언어가 브라우저에서 작동하지 못하게 오직 JavaScript 만 표준으로 정해놨다.
즉, 하나밖에 없으니깐, 불편해도 배워야한다.

그리고, 모든 브라우저에 기본적으로 내장되어 있다.
크롬 키고, 개발자도구 F12 눌러보자.

그리고 다음과 같이 입력하고 엔터 쳐보자.

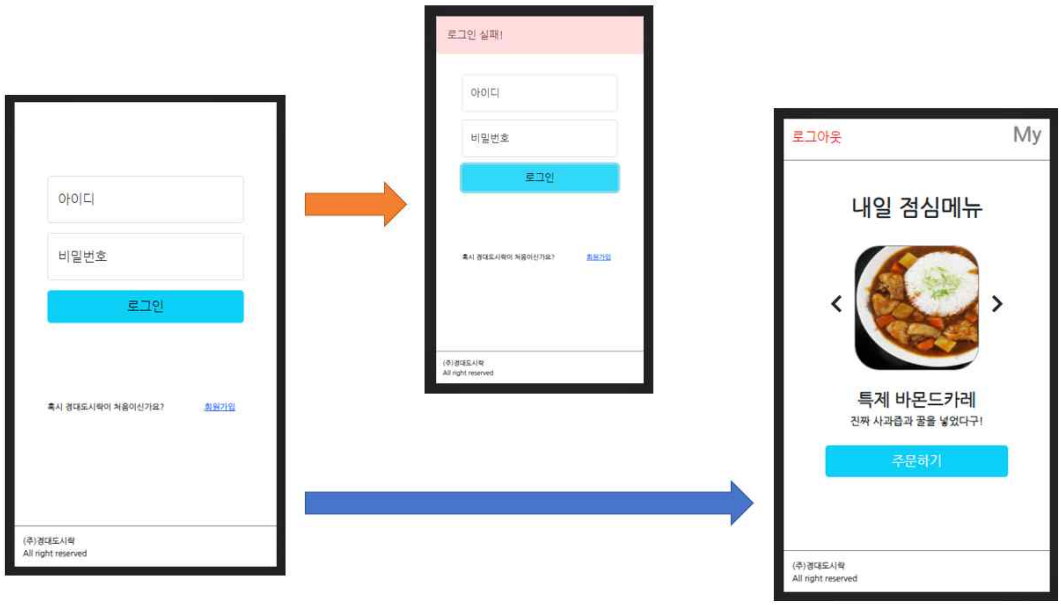
```
console.log("안녕 조교");
```



console.log() 는 print 라고 생각하면 된다. 테스트 할 때 엄청나게 많이 쓰는 기본이니 반드시 알아두자.

여기서 일일히 코드를 치진 않을것이고, VSCode 에서 미리 다 친 다음에 여기에 복사 붙여넣기할것이다.

JavaScript 가 있으면 뭘 할 수 있을까? 간단한 예를 통해 알아보면,



경대도시락 로그인 화면이다.
로그인을 일부러 실패해보면, 로그인 실패라는 알림이 상단에 1초 정도 떴다가 사라지고,
로그인 성공하면 메인화면으로 이동한다.
여러분이 HTML 과 CSS 만 가지고선 할 수 없는 작업이다.

버튼을 눌렀을 때 어떤 일이 일어난걸까?

1. 특정한 “함수”가 실행되고,
2. 여러분이 입력한 아이디와 패스워드가 실제 DB의 데이터와 일치하는지를 확인하기위해 서버로 아이디와 패스워드를 보낸다.
3. 사용자 정보가 일치하면,
   메인화면으로 화면 리다이렉트
   일치하지 않으면,
   알림을 1초동안 띄웠다가 사라지게 만듦

즉, JavaScript 로 웹 프로그래밍을 할 수 있다.

먼저 말하고 싶은 건, Java와는 완전히 다른 언어다.
어느 정도냐면, 인도와 인도네시아만큼이나 다르고
코끼리와 호랑이만큼이나 다르다.
단언컨데, 현재 메이저 언어,
Java, C, C++, Python, C# 중에서
JavaScript 와 비슷한 언어는 없다.
근데 왜 "Java"Script 냐고?
JavaScript 가 맨 처음 탄생했을 때 Java가 굉장히 유명했고
그 유명세를 따르려고 JavaScript 라고 이름 붙인 것.
실제로, JavaScript 의 정식 명칭은 ECMA Script 이다.

우리 수업 이름은 웹프로그래밍이고,
이것은 곧 여러분들이 JavaScript 에 익숙해지는것이 매우 중요하다는 뜻이다.
수많은 기업들이 웹 개발자를 뽑을 때,
여러분들이 JavaScript 를 잘 알고있는지를 평가할 것이고,
심지어 깨어있는 회사는 코딩테스트도 JavaScript "방식" 으로 낼 것이다.
여기서 "방식" 이라는 표현을 썼는데,
JavaScript는 언어 설계와 사용법이 매우 독특한 언어이기 때문에,
코딩 테스트 칠 때 주로 사용하는 C++, Java 와는 풀이방법이 완전히 다르고,
JavaScript 코딩테스트를 출제하는 출제자 역시 다른 언어들과는 다르게 풀기를 기대한다.

게다가, JavaScript는 2015년에 엄청나게 많은 부분이 버전업되었다.
흔히 ES6 문법이라고 부르는 것들이다.
기업들에서 웹개발자 면접을 보고 코딩테스트를 볼 때
여러분들이 2015년 이후에 문법을 당연히 알고 있을 것이라 가정하고
면접과 코딩테스트를 진행한다.

예를 들면, 면접 갔을 때 이런 문제를 낸다.
(본 조교가 직접 당한 비슷한 유형의 문제를 좀 더 쉽게 써봤다.
결국 그 회사 못갔다.)

ex) 언어: JavaScript
1부터 n까지의 수를 가지고 있는 배열이 있다.
[1, 2, 3, ..., n]
각각의 요소를 1씩 더해서 새로운 배열을 만들되,
반복문을 쓰지 말고, 재귀함수도 쓰지 말라.

이 경우, 출제자가 의도한 건, 콜백함수를 쓸 수 있는가, 더 정확하게는 map 함수를 쓸 줄 아느냐를 묻는 것이다.

sol)

```
const datas = [1, 2, 3, 4, 5];
const newDatas = datas.map((data) => data + 1);
console.log(newDatas);
```

결과:

```
[2, 3, 4, 5, 6]
```

자세히 보면, map 이라는 함수 안에 파라미터로 함수가 들어간다.
즉, JavaScript에서 함수는 곧 변수다.
다른 메이저 언어에선 함수는 변수가 아니다.
이런 형태를 여러분들은 따로 JavaScript 를 공부하지 않았다면 본 적이 없을 것이다.

그리고 JavaScript 에선 객체(object)라는 것을 일상적으로 쓸 것인데, 객체는 다른 언어들과는 다르게 클래스의 인스턴스가 아니고, 심지어 JavaScript에서 클래스는 다른 언어와 설계부터가 다르다.

여러분들은 1학년 때 C언어와 파이썬을 배웠고,
웹프로그래밍수업은 두 언어를 알고 있다고 가정하고 가르친다.
즉, 변수가 뭔지, 자료형이 뭔지에 대해 구구절절 설명하지 않는다.
기초가 안되어있는 학생들을 위해, 학습할 수 있도록 자료실에 교안을 올려두었다.
그리고 줌 강의는 항상 자료실에 링크를 통해 제공되고,
교안과 PPT, 코드를 업로드해주기때문에 복습을 꾸준히 해주길 바란다.

배워볼 내용은 다음과 같다.
1. 변수 let, const, var
2. 자료형(Type), 백틱(` `)
3. 조건문 if, else, ||, &&, ?, :
4. 배열(array)
5. 객체(object)

6. 함수 function, arrow function
7. 반복 callback, map(), filter()
8. 이벤트(Event) 와 돔(DOM) 조작
9. 비동기 통신(async)

여기서 추가로, JavaScript 실력을 향상시키기 위해 7가지 게임을 만드는 영상을 추후 자료실에 올려놓도록 하겠다.
게임을 줌 강의에서 직접 다루기 어려운 이유가 두 가지인데,

1. 난이도가 매우 높다.
2. 다뤄야 할 내용이 매우 많다.

이에, 정식 커리큘럼엔 포함시키지 않고 여러분들이 자습할 수 있도록 직접 영상을 찍어서 제공할 계획이다.

시작해보자.

1. 변수 const, let, var

JavaScript 엔 선언문이 존재한다. 변수를 쓰기 위해선 선언해야한다.
총 세 가지의 변수선언법이 있는데, 우선 결론만 말하겠다.

일단 const 를 사용하고, (90%의 상황이다)
const 를 사용할 수 없는 상황엔 let 을 사용하고,
var는 위험하다.

먼저 const 부터 살펴보자.

```
const sayHi = "안녕";
console.log(sayHi);
```

결과: 안녕

console.log() 는 프린트문이라고 생각하면 된다.
여러분이 마주칠 90% 의 상황에선, 변수 선언은 const 로 충분하며, 가장 안전하다.

여기서부턴 심화학습이다. 매우 어려운 얘기를 할 것이다.
왜 const를 써야하는지를 이론적으로 설명할 것이다.
const 의 핵심은, 변수를 못바꾼다는것이다.

```
> const sayHi = "안녕";
  sayHi = "헬로"; // error
✖ ▶Uncaught TypeError: Assignment to constant variable. VM24:2
      at <anonymous>:2:7
```

const는 constant의 줄임말이며, 즉 “상수” 이다.
그러나, 완전한 상수는 아니고 상수 변수(constant variable)인데, 다음의 예외때문이다.
첫번째. 배열 element 변경, 추가 및 삭제
두번째, 객체 property 변경, 추가 및 삭제
나중에 배우겠지만, 이게 가능하기 때문에 대부분의 선언, 90%의 상황은 const로 충분하다.

왜 변수를 안 바꾸는게 중요할까?
이것은 immutability, 불변성을 유지하기 위함이다.
불변성은, 여러분들이 포트폴리오 3개 이상 만들어본 중급개발자 이상이라면 관심있게 봐야 할 주제다.
웹 개발자는 서버로부터 들어오는 데이터든, 내가 브라우저에서 만든 데이터든, 그 데이터를 유저가 신뢰할 수 있도록 만들어야한다.
무슨 말이냐면, 웹 개발자는 변하지 않을 데이터와 변할 데이터를 구분해서 유저에게 제공해야한다.
여러분이 const를 쓰는 습관을 들이면, 여러분이 코딩하면서 겪는 실수가 줄어들뿐만 아니라, 해커의 공격으로부터도 여러분의 데이터를 보호해줄 것이다.

경대도시락의 메인페이지이다. 아랫쪽 (주)경대도시락 부분은 이 페이지뿐만 아니라 여러 페이지에서 쓰일 것이다.

이 정보가 매번 바뀐다면, (주)경대도시락 을 신뢰할 수 있겠는가?

인터넷 캡쳐 화면

실제 해킹당한 사이트를 한번 보자.
모 대통령 모교인 대구공고의 해킹사건이다.
디씨인사이드에서 이 사이트의 모든 데이터를 엉망으로 만들어놨다.
타이틀이 대구공고가 아니라 코갤고등학교로 바뀌어있다.

다시 정리해보면,
90% 의 변수선언은 const 로 충분하다.
그럼 나머지 10% 의 상황에선 무엇을 써야할까?

```
let sayHi = "안녕";
sayHi = "헬로";
console.log(sayHi);
```

결과: `'헬로'`

즉, let 을 사용하면 된다.
let 은 변수의 내용을 바꿀 수 있다.
단, 웹개발자가 let 을 쓸 때는, 반드시 필요해서 써야한다.

그리고 여러분들이 쓰지 말아야 할 변수선언법이 하나 있다.
2015년 이전에 쓰인 var 라는 변수선언법이다.

var는 선언된 변수를 다시 선언해서 쓸 수 있는데
이는 프로그래밍 언어학적으로 매우 위험하다.
그리고 여러분이 큰 기업 코딩테스트를 볼 때 var를 쓰면 떨어질 가능성이 크다.
기업들도 var의 위험성을 당연히 알고 있기 때문이다.

```
var sayHi = "안녕";
var sayHi = "헬로";
console.log(sayHi);
```

결과: `'헬로'`

언어학적으로, 한번 선언한 변수는 동일한 이름으로 다시 선언하면 안 된다!

근데 왜 여전히 JavaScript 에서 var를 제공할까?
이미 수억개의 사이트들이 var를 사용하여 코딩되었기때문.
var를 제거했을 때 생기는 인터넷 대혼란이 너무 크다.

결론을 다시 정리하면,
변수 선언시엔 일단 const를 사용하라. 90%의 상황엔 const 를 쓰면 된다.
const 를 사용할 수 없는 상황엔 let 을 사용하라.
var는 위험하다.

2. 자료형(type), 백틱(` `)

number - 숫자를 의미.
0 ,1, 2, -1, -2.58, ...
string - '실'이라는 뜻인데, '문자열'을 의미한다.
"a", "hello", 'TypeScript', '0'
큰따옴표로 쓰던, 작은따옴표로 쓰던 똑같이 string이다.
따옴표를 생략하면 키워드나 변수로 인식하므로, 변수를 string으로 쓰고싶으면
반드시 따옴표를 써야 한다.
중요! 0을 따옴표로 감싸면 number가 아닌 string으로 인식한다.
boolean - true 또는 false를 말한다. 참과 거짓.
따옴표를 써서 'true'라고 쓰면 boolean이 아닌 string으로 인식한다.

array, object, function - 역시 자료형이나, 난이도가 있는 주제이므로 각각의 장에서
따로 다룬다.

* 키워드(keyword)는 특정한 작업을 하는 단어들을 말한다. 우리가 지금까지 배운 키워드는
const, let, var이며, 앞으로 더 많은 키워드들을 배워볼 것이다.

typeof 를 통해 내가 쓴 자료형이 무엇인지 확인해보자.

먼저, 0과 "0"은 다르다.

```
const a = 0;
console.log(typeof a);  // number
const b = "0";
console.log(typeof b);  // string
```

비교연산자 === 로 확실히 알아보자.
비교연산자는 다음 장에서 좀 더 상세히 배울 것이다.
맞으면 true, 틀리면 false 반환한다.

```
console.log(0 === "0");  // false
```

string 은 항상 따옴표를 써줘야한다.
참고로, JavaScript 에선 String 이 아니라 string 이다.

```
const a = david;   // error
```

왜냐면, david는 string 이 아니라 하나의 변수로 인식하기 때문인데, 우린 david 라는 이름의 변수를 선언한 적이 없기 때문이다.
“david”라고 쓰면 정상작동한다.

다음, boolean을 알아보자. true 또는 false 를 말한다.

```
const a = true;
console.log(typeof a);  // boolean
```

하지만, “true”라고 쓰면 string 이다.

```
console.log(typeof "true"); // string
```

다음으로, 백틱 ` ` 에 대해 알아보자.
주로 문자열 안에서 변수나 함수 등을 좀 더 편하게 사용하려고 쓴다.
탭키 위에 작은따옴표 비슷하게 생긴 것이다.

백틱을 쓸 경우와 쓰지 않을 경우의 차이점을 보자.

JavaScript 에서 문자열 결합은 + 를 사용한다.

```
const myName = "조교행님";
const age = 28;

console.log(myName + "은 " + age + "살이다.");
```

결과:

```
조교행님은 28살이다.
```

백틱을 사용하면 다음과 같이 쓸 수 있다.

```
const myName = "조교행님";
const age = 28;

console.log(`${myName}은 ${age}살이다.`);
```

결과:

```
조교행님은 28살이다.
```

+ 기호를 사용하지 않고 문자열 결합을 할 수 있다.
백틱 안에 변수나 함수의 계산결과를 ${ } 를 사용해 쓸 수 있다.

백틱 안에선 엔터와 탭도 적용된다.
따라서, console.log() 를 여러 개 쓰고 싶지 않을 때 유용하다.

```
console.log("조교행님");
console.log("나이: 28세");
console.log("사는곳: 이마트 옆");
```

결과:

```
조교행님
나이: 28세
사는곳: 이마트 옆
```

```
console.log(`
    조교행님
    나이: 28세
    사는곳: 이마트 옆
`);
```

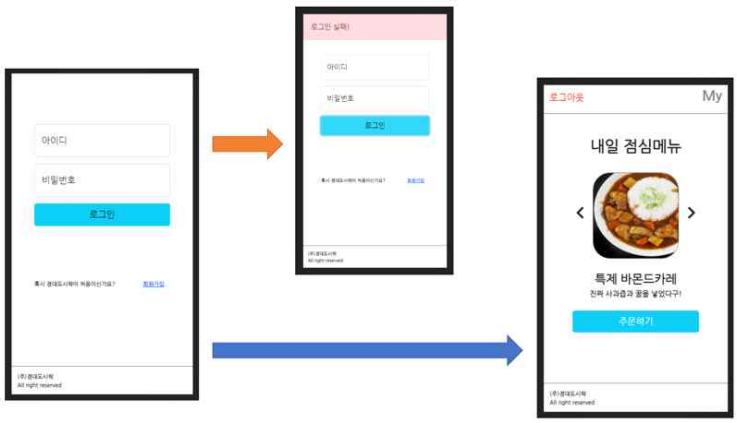
결과:

```

    조교행님
    나이: 28세
    사는곳: 이마트 옆
```

그러나, 엄밀히 말하면 결과는 다르다. 오른쪽 코드에선 엔터와 탭이 그대로 적용되어 한 줄의 결과로 출력된 것을 볼 수 있다.

3. 조건문 - if, else, ||, &&, ?, :

조건은 프로그래밍의 기본이다. 어느 정도는 알고 있겠지만, 복습하는 느낌으로 보자.



가장 흔히 활용되는 조건은 로그인이다.
아이디와 비밀번호가 일치하지 않으면 경고창을 띄우고,
일치하면 해당 사용자의 메인화면으로 넘어간다.

if else는 c언어와 똑같은데,
== 대신 === 를, != 대신 !== 를 사용하라

```
const myFavorite = "아빠";

if (myFavorite === "엄마") {
    console.log("엄마가 좋아");
} else if (myFavorite === "아빠") {
    console.log("아빠가 좋아");
} else if (myFavorite === "누나") {
    console.log("누나가 좋아");
} else {
    console.log("다 싫어");
}
```

왜냐면, == 또는 != 로만 쓰면 값 비교이기 때문이다.
=== 또는 !== 로 쓰면 데이터 타입까지 비교하므로 훨씬 정확하다.

```
console.log("0" == 0);  // true
console.log("0" === 0); // false

console.log("0" != 0);  // false
console.log("0" !== 0); // true
```

둘의 데이터타입은 string 과 number 이므로,
값 비교할때 쓰는 == 를 쓰면 true 이고
타입 비교까지 할때 쓰는 === 를 쓰면 false 이다.

대소비교 역시 가능하다.
>, <, >=, <= 를 사용할 수 있다.

```
const age = 20;

if (age > 20) {
    console.log("술 판매 가능");
} else {
    console.log("영업정지 삐용삐용");
}
```

결과:

```
영업정지 삐용삐용
```

만약, 20세도 포함하고 싶다면 > 대신 >= 를 쓰면 된다.

조건 안에 조건도 가능하다.

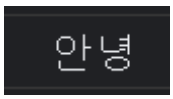
```
const myName = "조교행님";
const day = "일요일";

if (myName === "조교행님") {
    if (day === "일요일") {
        console.log("오늘은 출근 안함");
    } else {
        console.log("출근이다. 으악!");
    }
} else {
    console.log("누구냐 넌");
}
```

조건이 true일 경우를 고려해보자. 일단 다음은 기본이니 알고있을 것이다.

```
if (true) {
    console.log("실행");
}
```

그런데 다음도 가능하다.

```
const myName = "조교행님";
if (myName) {
    console.log("안녕");
} else {
    console.log("이름이 뭐야?");
}
```

결과: 

즉, JavaScript 는 뭐라도 들어가있으면 true 로 받아들인다.

다음은 false 로 인식한다.

```
const myName = "";
if (myName) {
    console.log("안녕");
} else {
    console.log("이름이 뭐야?");
}
```

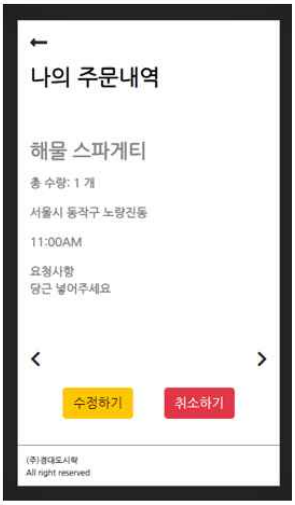
결과: 

빈 문자열이기 때문이다.

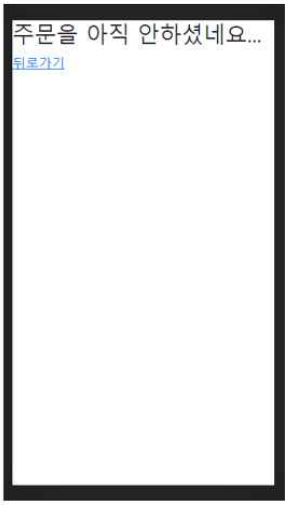
이것은 웹개발자라면 반드시 알고있어야 하는 특성이다.
현재 데이터 상태에 따라, 사용자에게 보여주는 화면을 다르게 만들 수 있기 때문이다.

경대도시락을 예로 들면, 주문내역이 있을 경우, 즉 데이터가 실제 존재할 경우와 데이터가 없을 경우에 보여주는 화면의 결과가 다르다.

주문내역이 있을 경우



주문을 아직 안 했을 경우



JavaScript 의 비교에서 false 로 인식하는 것들이 있다.

false	(boolean)
0	(number)
" "	(string. 빈 문자열)
undefined, null	(다음 장에서 배울 것)

추후 배우겠지만, 빈 배열 [ ] 과 빈 객체 { } 는 false 가 아니다.
위 언급한것들을 제외하곤 모두 true 이다.