😀

hoisting

❗ 호이스팅이란?

- 호이스팅은 코드를 실행하기 전 `변수선언/함수선언`을 해당 스코프의 최상단으로 끌어올리는 것이 아니다.
- 호이스팅은 코드가 실행하기 전 `변수선언/함수선언`이 해당 스코프의 최상단으로 끌어 올려진 것 같은 현상을 말한다.
- 자바스크립트 엔진은 코드를 실행하기 전 실행 가능한 코드를 형상화하고 구분하는 과정(*실행 컨텍스트를 위한 과정)을 거친다.
- 자바스크립트 엔진은 코드를 실행하기 전 실행 컨텍스트를 위한과정에서 모든 선언(var, let, const, function, class)을 스코프에 등록한다.
- 코드 실행 전 이미 `변수선언/함수선언`이 저장되어 있기 때문에 선언문보다 참조/호출이 먼저 나와도 오류 없이 동작한다.(정확히는 var 키워드로 선언한 변수와 함수 선언문일 경우 오류 없이 동작한다.이는 선언이 파일의 맨 위로 끌어올려진 것 처럼 보이게 한다.)
- 실행 컨텍스트는 실행 가능한 코드가 실행되기 위해 필요한 환경을 의미하고 실행되기 전 이러한 실행 컨텍스트 과정(코드를 구분하는 과정)을 거친다.

> 이 호이스팅이라는 용어를 자바스크립트 실행 컨텍스트에 의한 위에 설명한 현상을 호이스팅이라고 부른다는 것으로 이해하면 되겠다. 그 현상이란 선언이 코드 실행 보다 먼저 메모리에 저장되는 과정으로 인한 현상을 말한다.

❗ 변수 호이스팅 (var, let, const 키워드)

- 자바스크립트의 모든 선언에는 호이스팅이 일어난다.

- **그런데 let, const, class를 이용한 선언문을 호이스팅이 발생하지 않는 것처럼 동작한다.**
- **var 키워드로 선언된 변수와는 달리 let 키워드로 선언된 변수를 선언문 이전에 참조하면 참조 에러(ReferenceError)가 발생한다.**
- **이는 let 키워드로 선언된 변수는 스코프의 시작에서 변수의 선언까지 *일시적 사각지대(Temporal Dead Zone; TDZ)에 빠지기 때문이다.**

❗ 호이스팅 예시

👉 변수 선언에서의 호이스팅 예시

```
// 호이스팅 때문에 선언이 끌어올려져서 오류 안남.
console.log(text); // (선언 + 초기화 된 상태)
text = 'Hanamon!'; // (선언 + 초기화 + 할당 된 상태)
var text;
console.log(text);
```

문제

▼ **var 변수 선언과 함수선언문에서만 호이스팅이 일어난다.**

땡