단축 평가

논리 연산자를 사용한 단축 평가

논리곱(&&) 연산자는 두개의 피연산자가 모두 true로 평가될 때 true를 반환한다.

논리곱 연산자는 좌항에서 우항으로 평가가 진행된다.

```
//논리 연산의 결과를 결정하는 두 번째 피연산자, 즉 문자열 'Dog'를 그대로 반환한다.
'Cat' && 'Dog' // -> "Dog"
```

논리합(||) 연산자는 두 개의 피연산자 중 하나만 true로 평가되어도 true를 반환한다.

논리합 연산자도 좌항에서 우항으로 평가가 진행된다.

```
//논리 연산의 결과를 결정한 첫 번째 피연산자, 즉 문자열 'Cat'을 그대로 반환한다.
'Cat' || 'Dog' // -> "Cat"
```

논리곱(&&) 연산자와 논리합(||) 연산자는 논리 연산의 결과를 결정하는 피연산자를 타입 변환하지 않고 그대로 반환한다. 이를 단축 평가라 한다.

> 💡 단축 평가는 표현식을 평가하는 도중에 평가 결과가 확정된 경우 나머지 평가 과정을 생략하는 것

단축 평가 표현식	평가 결과
true || anything	true
false || anything	anything

```
// 논리합(||) 연산자
'Cat' || 'Dog'  // "Cat"
false || 'Dog'  // "Dog"
'Cat' || false  // "Cat"
```

단축 평가 표현식	평가 결과
true && anything	anything
false && anything	false

```
// 논리곱(&&) 연산자
'Cat' && 'Dog'  // "Dog"
false && 'Dog'  // false
'Cat' && false  // false
```

```
//어떤 조건이 참일 때 무언가를 해결해야 한다면 논리곱(&&)연산자 표현식으로 if문을 대체할 수 있다.
var done = true;
var message = '';

// 주어진 조건이 true일 때
if (done) message = '완료';

// if 문은 단축 평가로 대체 가능하다.
// done이 true라면 message에 '완료'를 할당
message = done && '완료';
console.log(message); // 완료

==========================================

//조건이 거짓일 때 무언가를 해야한다면 논리합(||) 연산자 표현식으로 if문을 대체할 수 있다.
var done = false;
var message = '';

// 주어진 조건이 false일 때
if (!done) message = '미완료';

// if 문은 단축 평가로 대체 가능하다.
// done이 false라면 message에 '미완료'를 할당
message = done || '미완료';
console.log(message); // 미완료

==========================================

//삼항조건 연산자로 if…else문을 대체할 수 있다.
```

```
var done = true;
var message = '';

// if...else 문
if (done) message = '완료';
else      message = '미완료';
console.log(message); // 완료

// if...else 문은 삼항 조건 연산자로 대체 가능하다.
message = done ? '완료' : '미완료';
console.log(message); // 완료
```

옵셔널 체이닝 연산자 (?.) - ES11(ECMAScript2020)에서 도입

좌항의 피연산자가 null 또는 undefined인 경우 undefined를 반환하고, 그렇지 않으면 우항의 프로퍼티 참조를 이어간다.

```
var elem = null;

// elem이 null 또는 undefined이면 undefined를 반환하고, 그렇지 않으면 우항의 프로퍼티 참조를 이어간다.
var value = elem?.value;
console.log(value); // undefined
```

예제) 변수의 값이 null 또는 undefined라도 오류를 발생 시키지 않게 작성하세요

```
var exam;
console.log(exam); // undefined
```

null 병합 연산자 (??) - ES11(ECMAScript2020)에서 도입

좌항의 피연산자가 null 또는 undefined인 경우 우항의 피연산자를 반환하고, 그렇지 않으면 좌항의 피연산자를 반환한다.

null 병합 연산자는 변수에 기본값을 설정할 때 유용하다.

```
// 좌항의 피연산자가 null 또는 undefined이면 우항의 피연산자를 반환하고, 그렇지 않으면 좌항의 피연산자를 반환한다.
var foo = null ?? 'default string';
console.log(foo); // "default string"
```

예제)결과를 맞춰 보세요

```
var foo1 = "abc" ?? 'default string';
console.log(foo); //

================================

var foo2 = undefined ?? 'default string'
console.log(foo);
```

결과