02

반복문

학습내용

01 for 반복문

02 while 반복문

03 for와 while문의 비교

04 do~while 반복문

05 break문

학습목표

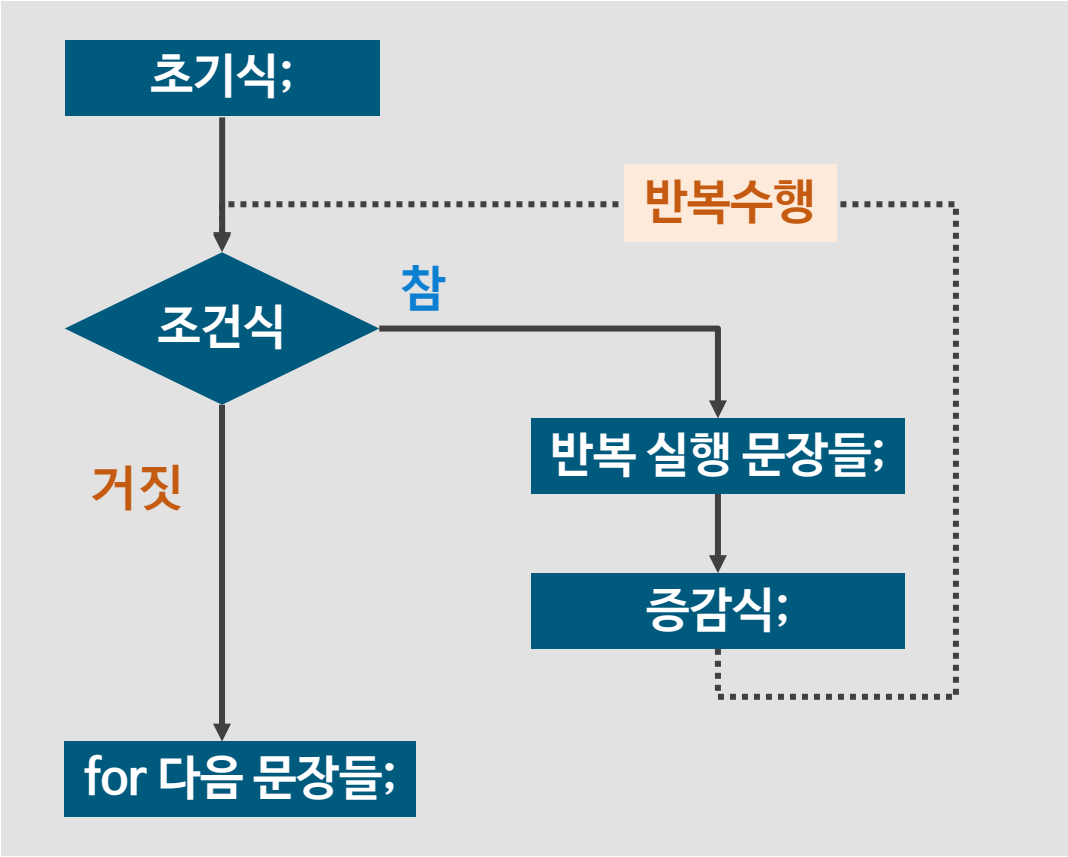
- for 반복문의 구조와 동작 원리를 설명하고, 다양한 상황에 맞게 반복문을 구현할 수 있다.
- while 반복문의 조건 평가 방식을 이해하고, 조건에 따라 반복을 제어하는 코드를 작성할 수 있다.
- for문과 while문의 차이점을 비교 분석하고, 목적에 따라 적절한 반복문을 선택하여 사용할 수 있다.
- do~while 반복문을 사용해 최소 한 번 이상 실행되어야 하는 반복 구조를 구현할 수 있다.
- break문을 활용하여 반복문 실행 흐름을 제어할 수 있다.

01

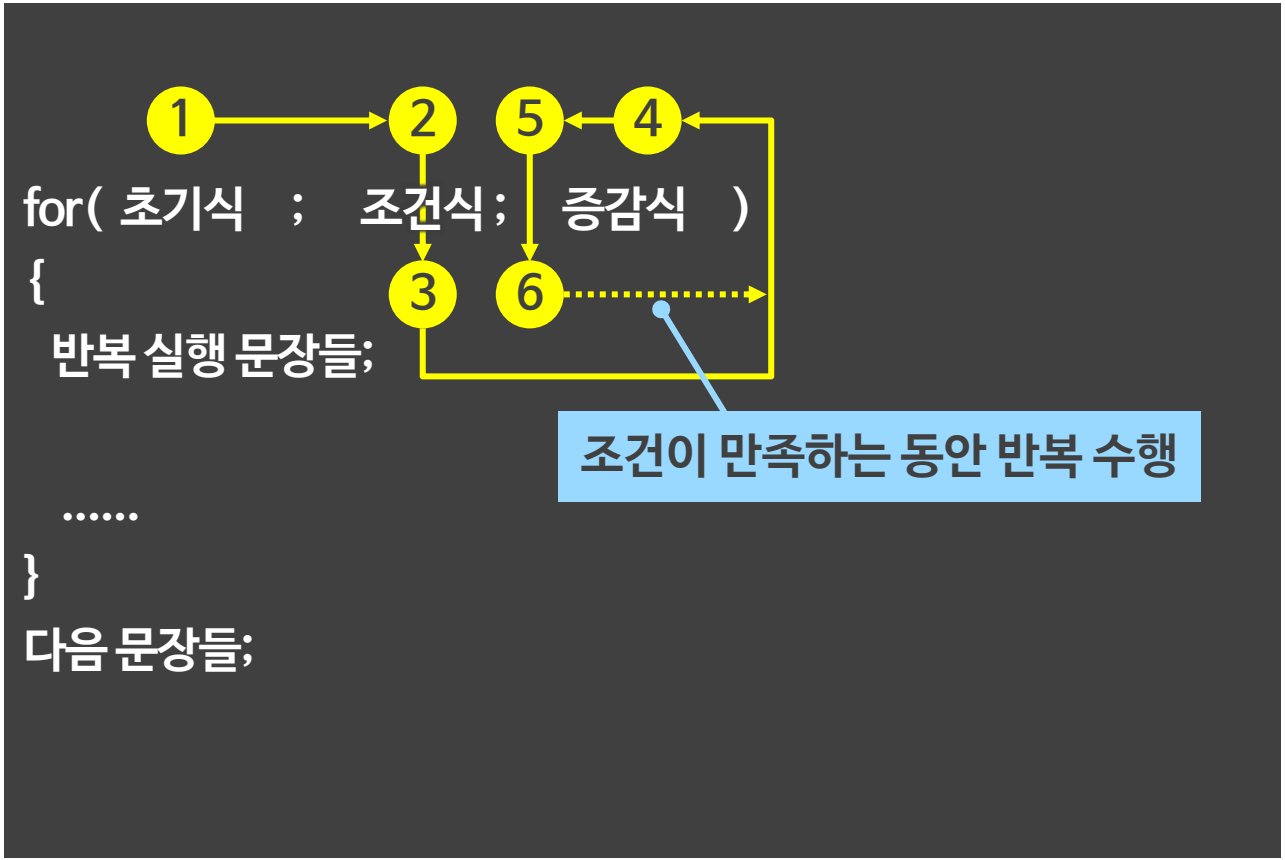
for 반복문

1) 사용법

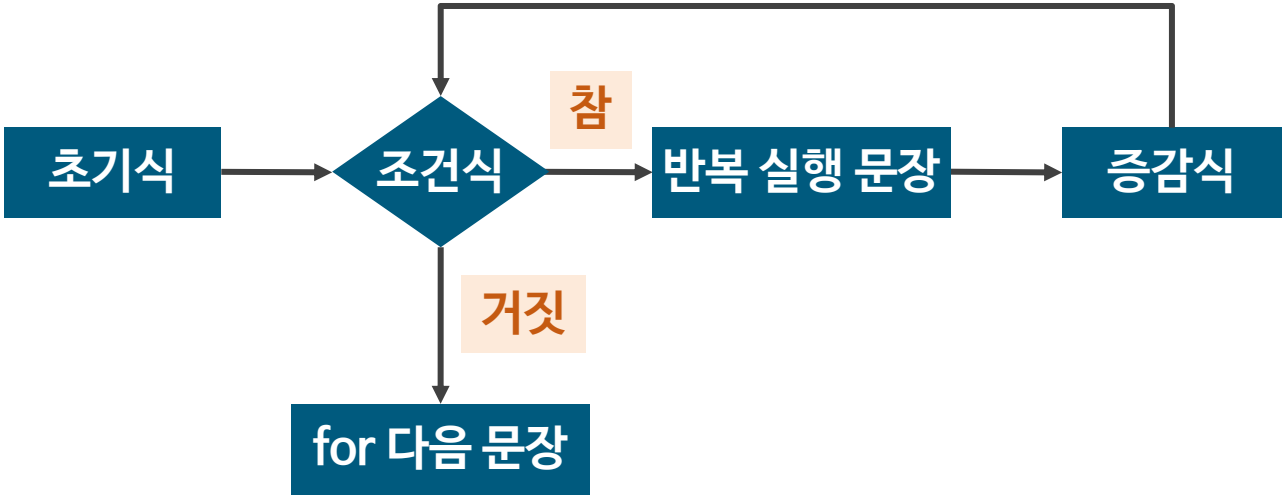

```
//초기식 : 변수 초기화
//조건식 : 반복 지속 여부 판단
//증감식 : 반복 시 변수 값 변화

for( 초기식 ; 조건식 ; 증감식 )
{
  반복 실행 문장들;

   ......
}
for 다음 문장들;
```



2) 수행 순서



2) 수행 순서



3) 무한루프


```
for(   ;    ;   )
{
   ......
}
```

탈출 조건이 없어, 항상 true이므로 무한 반복을 수행함

참고

```
for( 초기식 ; 조건식 ; 증감식 ) ;
```

for문 바로 뒤에 세미콜론(;)을 붙이면 실행할 반복문이 없다는 의미가 되므로 무의미한 반복이 이루어질 수 있음

4) 예시


```
let sum = 0;

for (let i = 1; i <= 10; i++) {
  sum += i;
}

console.log("for문 결과:", sum);
```

1~10까지 하나씩 증가해 가면서 반복적으로 누적을 구해 총 합계를 구함

```
[실행결과]
for문 결과 : 55
```

5) 확장 for문

멀티 패러다임 언어이기 때문에 배열, 객체, 이터러블, 함수형 프로그래밍 등
다양한 상황에 맞게 반복을 처리할 수 있는 여러 문법 제공

종류	설명	예시
for 기본형	정해진 횟수만큼 반복	`for (let i = 0; i < 5; i++) { ... }`
for...in [ES3]	객체의 key(속성 이름)를 순회	`for (let key in obj) { console.log(key); }`
for...of [ES6]	배열/반복 가능한 객체(Iterable)의 값을 순회	`for (let value of arr) { console.log(value); }`
forEach() [ES5]	배열의 각 요소에 대해 함수 실행	`arr.forEach((item) => { console.log(item); });`

참고

- 명령형 스타일 : for, for...in, for...of
- 함수형 스타일 : forEach(), map(), filter(), reduce()

- 객체 순회 예시

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

for (let key in person) {
  console.log(key);
                        // “name”, “age”
  console.log(person[key]);
                        // “Alice”, 25
}
```

객체의 key 값을
순회 ➡ 배열에 쓰면
index(문자열 형태)를
반환하므로 부적합

- 배열 순회 예시

```
let sum = 0;
let numbers = [1,2,3,4,5,6,7,8,9,10];

for (let num of numbers) {
  sum += num;
}
console.log("1부터 10까지의 합:", sum); // 출력: 55
```

1~10까지
하나씩 증가해 가면서
배열의 값을 순회

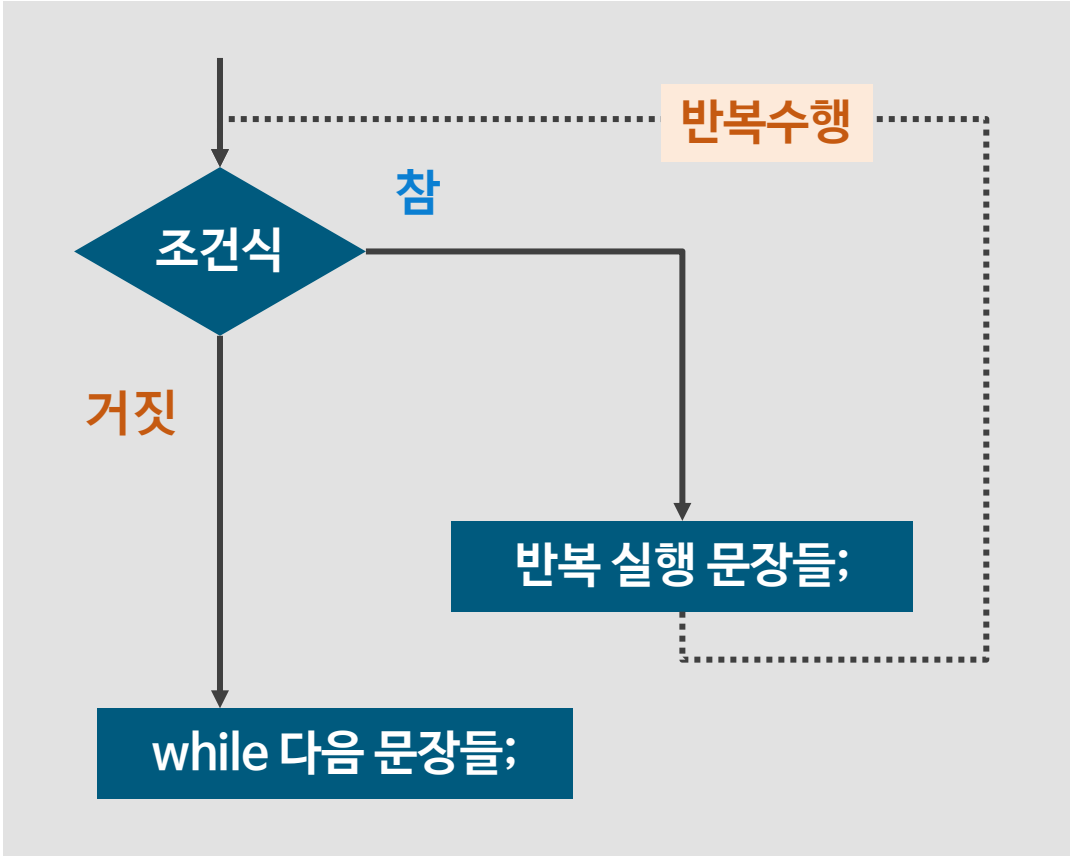
```
[실행결과]
for문 결과 : 55
```

02

while 반복문

1) 사용법


```
while( 조건식 )
{
  반복 실행 문장들;

   ......
}
while 다음 문장들;
```



2) while문 사용 예시


```
let i = 1;
let sum = 0;

while (i <= 10) {
  sum += i;
  i++;
}

console.log("while문 결과:", sum);
```

1~10까지
하나씩 증가해 가면서
반복적으로 누적을 구해
총 합계를 구함

[실행결과]

```
while문 결과 : 55
```

3) 무한루프


```
while( true )
{
    ......
}
```

탈출 조건이 없고, 항상 true이므로
무한 반복을 수행함

3) 무한루프


```
while( true ) ;
{
   ......
}
```

- 종료 기호를 잘못 기재하면 아래 블록은 하나의 블록으로 인식되고, while문은 단순히 조건식이 true인 무한 루프로 인식
- 여기서는 while 조건식이 항상 true가 되어 while문을 탈출하지 못해 그 이후 { } 문장들을 수행하지 못함

- while문과 상관없는 블록 요소임
- while문을 탈출한 이후 수행될 문장이 됨

03

for와 while문의 비교

1) 주요 사용 방법

For문

- **반복 횟수가 명확**할 때 가장 많이 사용
- 구조가 간결하고 직관적
  - 초기화, 조건, 증가 모두 한 줄

while문

- 반복 조건이 상황에 따라 변하거나 **언제 끝날지 모를 때 유리** (예 사용자 입력, 서버 응답 대기 등)

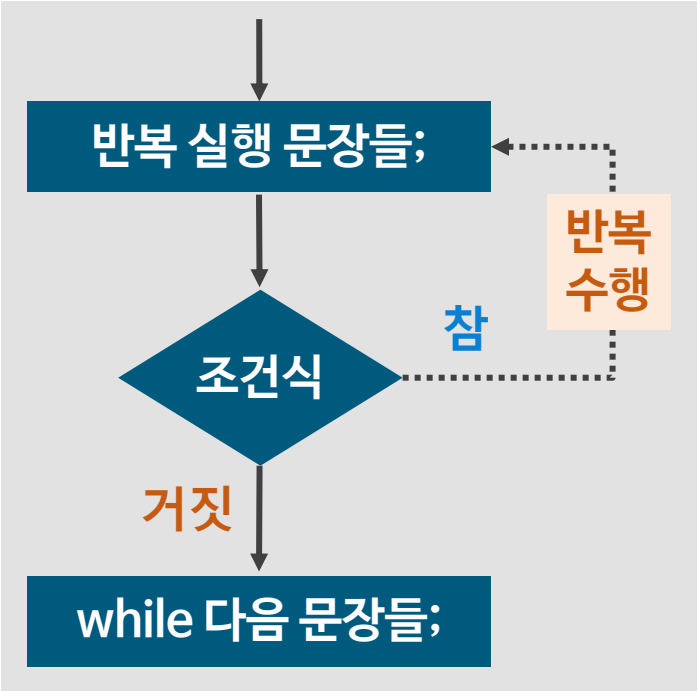
```
while(true) {
        if(특정조건)
            break;
}
```

04

do~while 반복문

1) do-while문


```
do {

  반복 실행 문장들;

   ……

} while( 조건식 );

do-while 다음 문장들;
```



반복 실행할 문장들을 먼저 수행하고,
조건 검색을 차후에 확인한 후, 반복을 결정함

2) while문 사용 예시


```
let i = 1;
let sum = 0;

do {
  sum += i;
  i++;
} while (i <= 10);

console.log("do...while문 결과:", sum);
```

조건 검사 전에
최소 한 번은 실행되는
반복문을 수행하여
총 합계를 구함

[실행결과]

```
do...while문 결과 : 55
```

05

break문

1) switch, while, for문의 블록 탈출

break문은 반복문(for, while)이나 switch문을
즉시 종료시키는 명령

```
for (let i = 1; i <= 10; i++) {
  if (i === 6) {
            // i가 6이 되면 반복 종료
            break;
  }
  console.log(i);
}
```

if 블록이 아닌(if 블록은 무시) 가장 가까운 for 블록문을 탈출

[실행결과]
1 2 3 4 5

세종사이버대학교

03주. 흐름을 제어한다! 제어문

03

제어문 활용 실습