● **반복문:** 어떤 작업이 반복적으로 수행될 때 사용되며, for문과 while문, 그리고 do-while문이 있다. 반복문은 주어진 조건을 만족하는 동안 주어진 문장들을 반복적으로 수행하므로 조건식을 포함한다. for문과 while문에 속한 문장은 조건에 따라 한 번도 수행되지 않을 수 있지만, do-while문에 속한 문장은 무조건 최소한 한 번은 수행될 것이 보장된다. for문과 while문 중 어떤 반복문을 선택해도 괜찮지만, for문은 주로 반복 횟수를 알고 있을 때 사용한다.

1. for문

- 조건이 참일 동안 블러{} 내의 문장들을 반복하다 거짓이 되면 반복문을 벗어난다.
- 순서: 1) 초기화 2) 조건식-조건식이 참일 동안 3) 수행될 문장 4) 증감식
- 초기화: 둘 이상의 변수가 필요할 때는 콤마(',')를 구분자로 변수를 초기화 하면 되며, 두 변수의 타입은 같아야 한다.
- 증감식: 증감 연산자 ++, --, 복합 대입 연산자 +=, *= 등 사용가능하다. 쉼표(',')를 이용해서 두 문장 이상을 하나로 연결해서 쓸 수 있다.

```
for (int i = 1, j = 10; i <= 10; i++, j--) { .... }
```

- 초기화, 조건식, 증감식은 생략 가능하며, 세 가지 모두를 생략할 경우 조건식은 참이 되어 무한 반복문이 된다. 대신 블럭{} 안에 if문을 넣어서 특정 조건을 만족하면 for문을 빠져 나오게 해야 한다.

```
for (;;) { .... }
// 나머지 연산자(%)를 사용하면 특정 범위의 값들이 순환하면서 반복되는 결과를 얻을 수 있다. // 나누기 연산자(/)는 같은 값이 연속적으로 반복되게 할 수 있다.
```

2. **중첩 for문:** for문의 블럭 안에 다든 for문을 포함시키는 것.

- 만약 중첩 for문이 3개 이고, 각 반복문이 3번씩 반복하면, 모두 (3*3*3=27) 27번이 반복된다.

3. 향상된 for문 (enhanced for statement)

```
for (타입 변수명 : 배열 또는 컬렉션) { // 반복할 문장 }
```

- 타입은 배열 또는 컬렉션의 요소의 타입이어야 한다.
- 배열 또는 컬렉션에 저장된 값이 매 반복마다 하나씩 순서대로 읽혀서 변수에 저장된다. 그리고 반복문의 괄호{} 내에서는 이 변수를 사용해서 코드를 작성한다.
- 향상된 for문은 배열이나 컬렉션에 저장된 요소들을 읽어오는 용도로만 사용할 수 있다는 제약이 있다.

```
// ex) int[] arr = {1, 2, 3, 4, 5} 배열의 모든 요소 차례대로 읽어오기. // 1. for문 for(int i = 0; i < arr.length; i++) {
System.out.println(arr[i]); } for(int num : arr) { System.out.println(num); }
```

Resource: 자바의 정석