04장(조건문과 반복문)

01. 조건문

if 문

```
if (조건식) {
        // 조건식이 참(true)일 때 수행될 문장들을 적는다.
}
```

조건식	조건식이 참일 조건
90 <= x && x <= 100	정수 x가 90이상 100이하일 때
x < 0 || x > 100	정수 x가 0보다 작거나 100보다 클 때
x%3==0 && x%2!=0	정수 x가 3의 배수지만, 2의 배수는 아닐 때
ch=='y' || ch=='Y'	문자 ch가 'y' 또는 'Y'일 때
ch==' ' || ch=='\t' || ch=='\n'	문자 ch가 공백이거나 탭 또는 개행 문자일 때
'A' <= ch && ch <= 'Z'	문자 ch가 대문자일 때
'a' <= ch && ch <= 'z'	문자 ch가 소문자일 때
'0' <= ch && ch <= '9'	문자 ch가 숫자일 때
str.equals("yes")	문자열 str의 내용이 "yes"일 때(대소문자 구분)
str.equalsIgnoreCase("yes")	문자열 str의 내용이 "yes"일 때(대소문자 구분안함)

if - else 문

```
if (조건식) {
        // 조건식이 참(true)일 때 수행될 문장들을 적는다.
} else {
        // 조건식이 거짓(false)일 때 수행될 문장들을 적는다.
}
```

if - else if 문

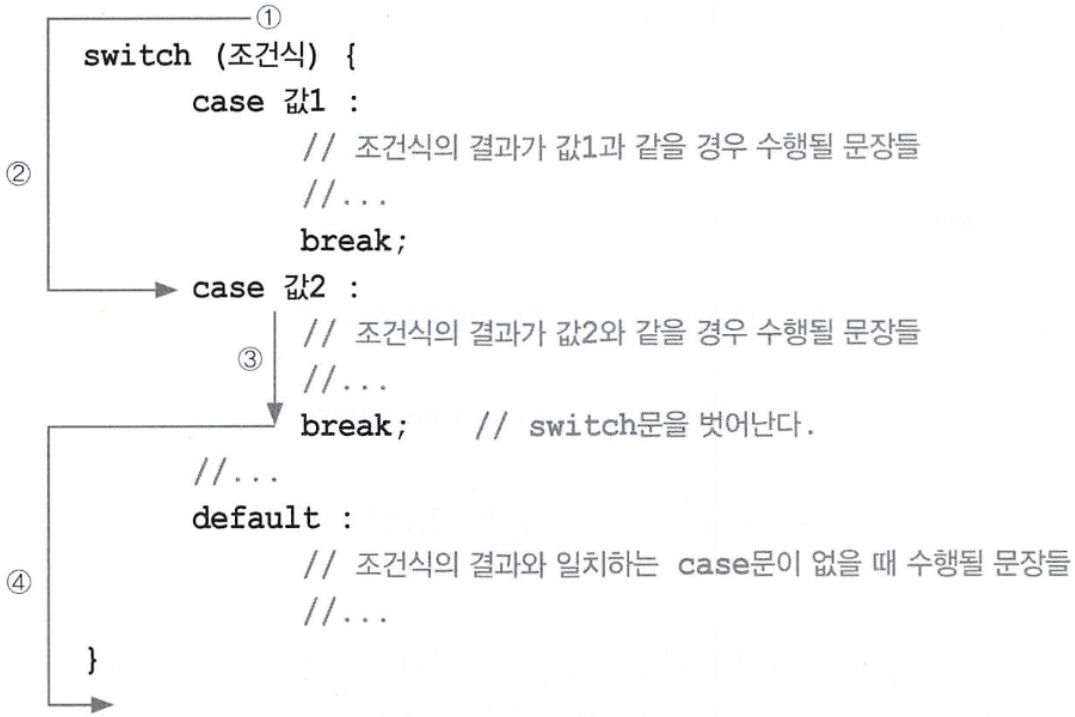
```
if (조건식1) {
        // 조건식1의 연산결과가 참일 때 수행될 문장들을 적는다.
} else if (조건식2) {
        // 조건식2의 연산결과가 참일 때 수행될 문장들을 적는다.
} else if (조건식3) {          // 여러 개의 else if를 사용할 수 있다.
        // 조건식3의 연산결과가 참일 때 수행될 문장들을 적는다.
} else {  // 마지막에는 보통 else블럭으로 끝나며, else블럭은 생략가능하다.
        // 위의 어느 조건식도 만족하지 않을 때 수행될 문장들을 적는다.
}
```

if 문은 중첩사용이 가능

```
if (조건식1) {
        // 조건식1의 연산결과가 true일 때 수행될 문장들을 적는다.
        if (조건식2) {
                // 조건식1과 조건식2가 모두 true일 때 수행될 문장들
        } else {
                // 조건식1이 true이고, 조건식2가 false일 때 수행되는 문장들
        }
} else {
        // 조건식1이 false일 때 수행되는 문장들
}
```

switch 문

- 처리할 경우의 수가 많은 경우에는 if문 보다 switch문으로 작성하는 것이 옳다.
- switch문의 조건식 결과는 정수 or 문자열 이어야한다.
- case문의 값은 정수 상수만 가능하며, 중복되지 않아야 한다.(결과값이 1:1 매칭이 되어야함)



① 조건식을 계산한다.
② 조건식의 결과와 일치하는 case문으로 이동한다.
③ 이후의 문장들을 수행한다.
④ break문이나 switch문의 끝을 만나면 switch문 전체를 빠져나간다.

---

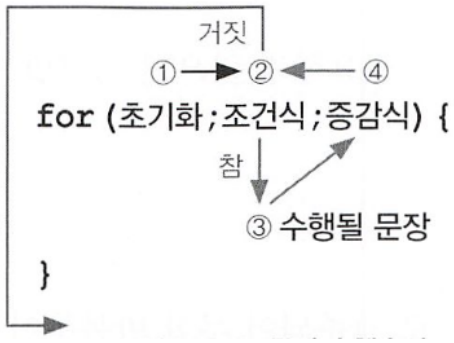
02. 반복문

for 문

- 초기화 → 반복문에 사용될 변수를 초기화하는 부분이며 처음에 한번만 수행
- 조건식 → true면 반복하고 false면 중단

- 증감식 → 제어하는 변수의 값을 증가 or 감소시키는 식

```
for (초기화;조건식;증감식) {
    // 조건식이 참일 때 수행될 문장들을 적는다.
}
```



중첩 for 문

- for문 안에 또 다른 for문을 포함시키는 것

향상된 for 문

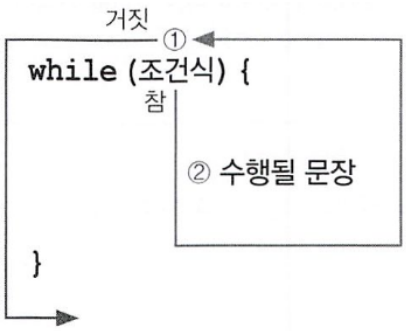
- 배열과 컬렉션에 저장된 요소를 접근할 때 기존보다 편리한 방법으로 처리하는 방법(?)

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

while 문

- while문의 조건식은 생략이 불가능

```
while (조건식) {
    // 조건식의 연산결과가 참(true)인 동안, 반복될 문장들을 적는다.
}
```



do - while 문

- 기본적인 구조는 while문과 같으나 조건식과 블럭{ }의 순서를 바꿔놓은 것
- 조건식이 맞지 않더라도 최소 do블럭을 한번 수행함

```
do {
        // 조건식의 연산결과가 참일 때 수행될 문장들을 적는다.
} while (조건식); ← 끝에 ';'을 잊지 않도록 주의
```

break 문

- 조건 or 반복문을 빠져나오게 하는 명령어

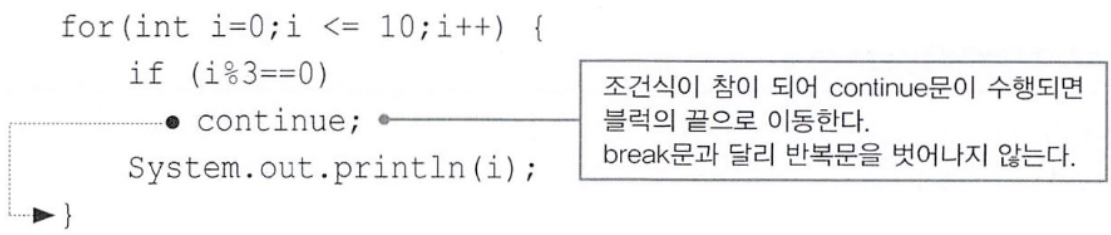
```
while(true) {
    if(sum > 100)
        break;
    ++i;
    sum += i;
} // end of while
```

break문이 수행되면 이 부분은 실행되지 않고 while문을 완전히 벗어난다.

continue 문

- 반복문 내에서만 사용

- 해당 예약어를 만나면 해당 반복문의 마지막 블록으로 이동(중간 스킵)



이름 붙은 반복문

- break문은 근접한 단 하나의 반복문만 벗어 날 수 있기에 특정 이름을 붙여 그 반복문을 벗어나해줌
- continue문에도 사용 가능