

조건문과 반복문



- 조건문
- 반복문
- break
- continue
- return

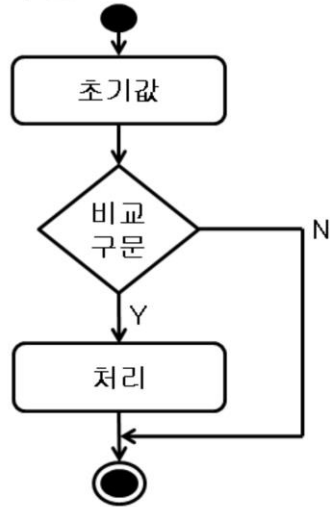
조건문

- 조건이 만족하면 프로그램 실행

	조건 대상	조건 내용
if/else	범위	규칙적인 조건에 따른 공통 처리 소수, 홀수, 짝수(연산 처리)
switch/ case	값	불 규칙적인 조건에 따른 공통 처리 메뉴, 상태(state), 플래그(flag) 체크

조건문

- if 구문



```
[초기값;]  
if(초기값 비교){  
    //처리  
}
```

조건문

- 실습) 9시 부터는 수업시간, 이전은 휴식시간

>java IfTest_1 10

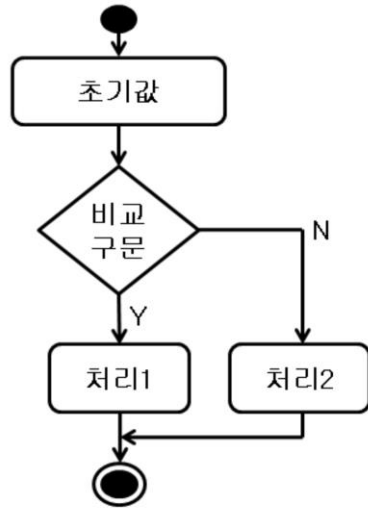
```
package com.oopsw.condition;
public class IfTest_1 {
    public static void main(String[] args) {
        //args[0]String ==>int
        int time=Integer.parseInt(args[0]);

        if(time>=9){
            System.out.println("수업시간");
        }else{
            System.out.println("휴식시간");
        }
    }
}
//main end
}
//IfTest_1 end
```

4-4

조건문

- if /else구문



```
초기값;  
if(초기값 비교){  
    //처리1  
}else{  
    //처리2  
}
```

조건문

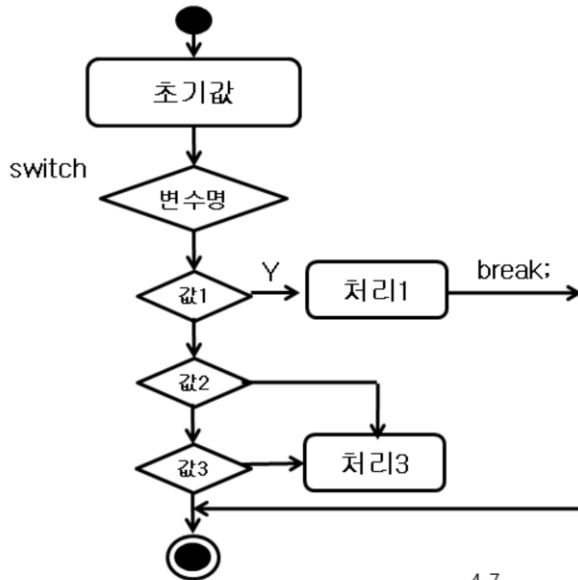
- 실습) 중첩 조건문

>java IfTest_2 10

```
//args[0]String ==>int
int time=Integer.parseInt(args[0]);
String message="시간 형식은 0~24 이전까지 입니다.";
if(time>=0 && time<24){
    if(time>=9 && time<12) message="오전수업";
    else if(time>=12 && time<13) message="점심";
    else if(time>=13 && time<18) message="오후수업";
    else
        message="휴식";
    message=time+"시는 "+ message+"시간 입니다";
}
//전체 if end
System.out.println(message);
```

조건문

● switch 구문



```
//byte, short, int, char, String"  
int num=10; //초기값  
switch(변수명){  
    case 값1 :  
        //처리1  
        break;  
    case 값2 :  
    case 값3 :  
        //처리3  
        break;  
    default :  
}
```

4-7

조건문

● 실습) 메뉴 구분하는 출력

```
int menu=Integer.parseInt(args[0]);
String message="";

switch (menu) {
    case 1: message="일번"; break;
    case 2: message="이번"; break;
    case 5: message="오번"; break;
    case 9: message="구번"; break;

    default : message="정의되지 않은 ";
}
message=menu+"는 "+ message+" 메뉴 입니다";
System.out.println(message);
```

실행결과

```
>java SwitchTest 5
5는 오번 메뉴 입니다
```

```
>java SwitchTest 11
11는 정의되지 않은 메뉴 입니다
```


조건문

- 실습)월에 따른 일수 출력

```
int month=Integer.parseInt(args[0]);
String message="월 형식은 1~12까지 입니다.";
if(month>=1 && month<=12){
    if(month==2 )
        message="28";
    else if(month==4 || month==6 || month==9 || month==11)
        message="30";
    else
        message="31";
    message=month+"월은 "+ message+"일까지 입니다";
}
System.out.println(message);
```

조건문

- 실습)월에 따른 일수 출력

```
int month=Integer.parseInt(args[0]);
String message="월 형식은 1~12까지 입니다.";
if(month>=1 && month<=12){
    switch (month) {
        case 2: message="28"; break;
        case 4:
        case 6:
        case 9:
        case 11: message="30"; break;
        default : message="31";
    }
    message=month+"월은 "+ message+"일까지 입니다.";
}
System.out.println(message);
```

4-10

//1.입력

```
Scanner sc=new Scanner(System.in);
System.out.print("월:");
int month=sc.nextInt();
String message="월 형식은 1~12까지 입니다.";
if(!(month>=1 && month<=12)){
    System.out.println(message);
    return;
}
switch (month) {
    case 2: message="28"; break;
    case 4:
    case 6:
    case 9:
    case 11: message="30"; break;
    default : message="31";
}
//switch
System.out.println(month+"월은 "+ message+"일까지 입니다.");
```

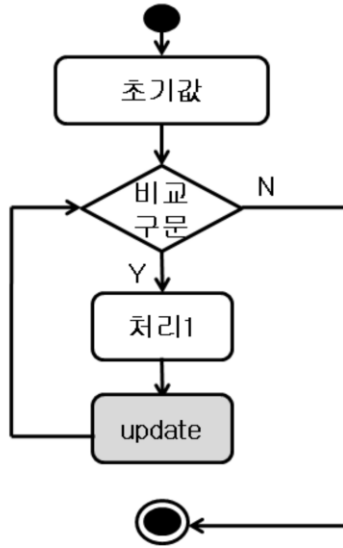
반복문

- 조건이 만족하는 동안 반복적으로 프로그램 실행

	반복횟수	처리 규칙
for	0~무한대	반복횟수가 예측이 가능하거나 정해진 반복횟수 처리시 사용 ex) 배열
while		반복 횟수가 예측 불가능하거나 무한 반복 처리시 사용 ex) 서버 프로그램, 검색 결과
do/while	1~무한대	

반복문

- for / while



```
for(초기값; 비교; update){  
    //처리1;  
}
```

```
for( ;true; ){  
}
```

```
초기값;  
while(비교){  
    //처리1;  
    update;  
}
```

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

4-12

반복문

- 실습)for문

```
for(int i=1; i<=10; i++){  
    System.out.println(i);  
}//for end  
System.out.println("while");
```

반복문

- 실습)while문

```
int j=0;
while(j<10){
    System.out.println(++j);
}

int k=0;
while(true){
    if(k++ ==10 ) break;
    System.out.println("k:" + k);
}
```

반복문

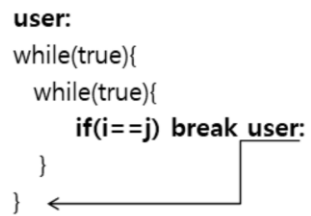
- 실습)do while문

```
int m=0;
do{
    System.out.println(++m);
}while(m<10);
```

break

- break 키워드 사용
 - switch 문을 빠져 나올때
 - 하나의 반복문을 빠져 나올때
 - 중첩된 반복문에서 특정 위치를 빠져 나올때

```
user:
while(true){
  while(true){
    if(i==j) break user:
  }
} ←
```

A diagram illustrating the use of the 'break' keyword in a nested loop. The code shows an outer loop labeled 'user:' and an inner loop. Inside the inner loop, there is an 'if' statement: 'if(i==j) break user:'. A line connects the 'break user:' statement to the 'user:' label, indicating that the loop will exit to the point just before the 'user:' label.

continue

- **continue 키워드 사용**
 - 현재 키워드가 있는 위치의 반복문 처음으로 이동할 때
 - 중첩된 반복문에서 특정 위치로 이동해서 시작할 때

```
user: <
while(true){
    while(true){
        if(i==j) continue user:
    }
}
```

return

- **return 키워드 사용**
 - 리턴 타입이 있는 메서드에서 값을 전달하고자 할때
 - 리턴 타입이 없어도 현재의 메서드를 종료하고자 할때

```
public void methodA(int num)
{
    if(num==0) return;
    System.out.println(num);
} ←
```

LAB

- 1부터 10까지 중에서 짝수의 수 합을 구하는 프로그램을 작성
- 입력한 숫자를 이용해서 구구단을 작성

입력 : 3

$3*1=3$

$3*2=6$

...

$3*9=27$

while / for 를 이용해서 각각 작성