

컴퓨터 프로그래밍 (Computer Programming)

이 선 순



9. 반복문의 심화



목차

1. while문

2. do~while문

3. 기타 제어문

03

기타 제어문

03 기타제어문 : break문

■ 반복문을 탈출하는 break문

- 지금까지 배운 반복문을 빠져나가는 방법은 조건식의 결과가 거짓이거나 사용자가 직접 [Terminate] 버튼을 누를 때임. 이외에도 반복문을 논리적으로 빠져나가는 방법인 break 문이 있음
- for문, while문, do~while문의 실행을 중지할 때 사용됨.
- switch문에서도 break문을 사용하여 종료함

```
반복문(for, while, do~while)  
{
```

...

```
break;
```

...

```
}
```

무조건 반복문
블록 밖으로 탈출

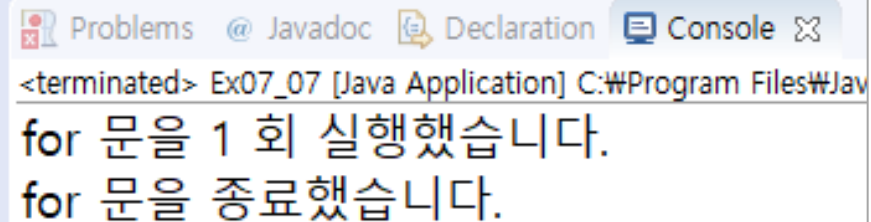


그림 7-11 break 문의 작동

03 기타제어문: break문

■ Break문 사용

```
1 //실습7-7. break문 사용 예1
2 public class Ex07_07 {
3     public static void main(String[] args) {
4         int i;
5
6         for (i = 1; i <= 100; i++) {           //100번 반복
7             System.out.printf("for 문을 %d 회 실행했습니다.\n", i); //변수 i 번째 출력
8             break;                             // 무조건 for문을 빠져나감
9         }
10
11         System.out.printf("for 문을 종료했습니다.\n");
12     }
13 }
```



<terminated> Ex07_07 [Java Application] C:\Program Files\Java\ for 문을 1 회 실행했습니다.
for 문을 종료했습니다.

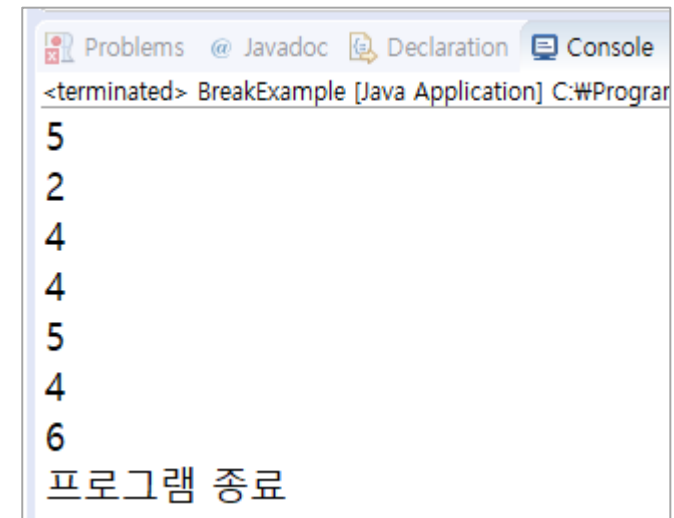
- 8행의 break문이 없다면 무조건 100번을 출력하는 프로그램
- break가 있어서 7행을 한번만 실행한 다음 for문을 빠져나가 10행이 비어있으므로 11행을 마지막으로 7실행함

03 기타제어문 : break문

■ Break문 사용

- Break문은 무한루프안에서 if문과 함께 사용되는 경우가 많음 : 무한루프를 돌다가 특정조건이 되면 빠져나가도록 할 때 break문을 사용함

```
1 //예1. break로 while문 종료|
2 package sec02.exam08;
3
4 public class BreakExample {
5     public static void main(String[] args) throws Exception {
6         while(true) {
7             int num = (int)(Math.random()*6) + 1;
8             System.out.println(num);
9             if(num == 6) {
10                 break;
11             }
12         }
13         System.out.println("프로그램 종료");
14     }
15 }
```



<terminated> BreakExample [Java Application] C:\WProgar

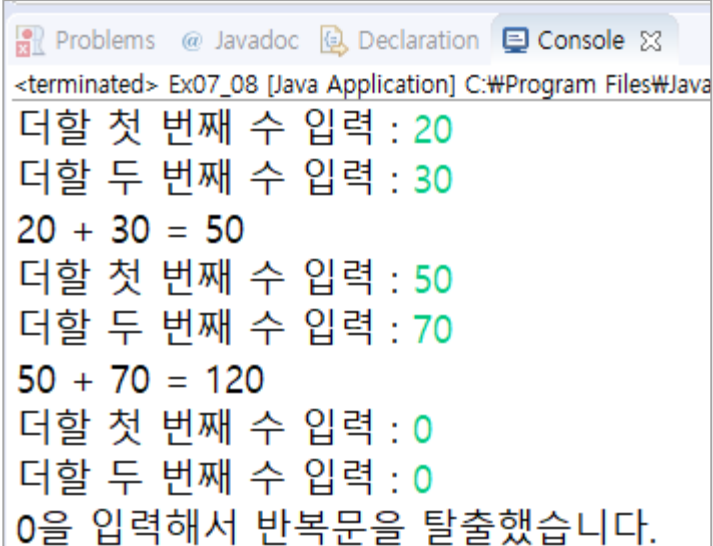
5
2
4
4
5
4
6
프로그램 종료

- Math.random() : 0~1 사이에서 균일한(uniform) 난수를 발생하는 함수

03 기타제어문 : break문

```
1 //실습7-8 break문 사용 예2
2 import java.util.Scanner;
3
4 public class Ex07_08 {
5     public static void main(String[] args) {
6         Scanner s = new Scanner(System.in);
7         int a, b;
8
9         while (true) {
10             System.out.printf("더할 첫 번째 수 입력 : ");
11             a = s.nextInt();    // 숫자를 입력받는다
12             System.out.printf("더할 두 번째 수 입력 : ");
13             b = s.nextInt();    //숫자를 입력받는다
14
15             if (a == 0)          //첫번째 입력값이 0이면
16                 break;          //무조건 while문을 빠져나간다
17
18             System.out.printf("%d + %d = %d \n", a, b, a + b);
19         }
20
21         System.out.printf("0을 입력해서 반복문을 탈출했습니다.\n");
22     }
23 }
```

- [실습7-3]을 break문을 사용하여 수정,
- 첫번째 숫자 0이 입력될 때 자동으로 종료
- 11행에서 입력된 값(변수a)에 0을 넣으면 15행의 'a==0'은 참이 되고, 16행의 break문을 만난 뒤 21행으로 이동하여 반복문을 탈출함



```
<terminated> Ex07_08 [Java Application] C:\Program Files\Java
더할 첫 번째 수 입력 : 20
더할 두 번째 수 입력 : 30
20 + 30 = 50
더할 첫 번째 수 입력 : 50
더할 두 번째 수 입력 : 70
50 + 70 = 120
더할 첫 번째 수 입력 : 0
더할 두 번째 수 입력 : 0
0을 입력해서 반복문을 탈출했습니다.
```


03 기타제어문 : break문

■ 실행 문장이 하나일 때의 블록 사용

- [실습 7-8]의 15, 16행은 다음 표현 방법 중 어떤 것을 사용해도 된다.

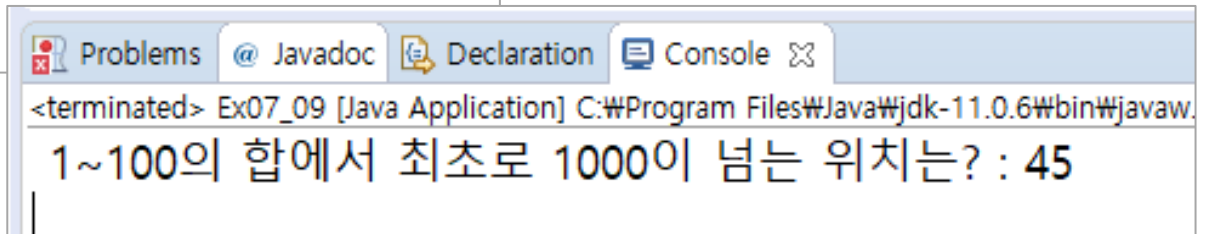
<pre>if(a==0) break;</pre>	==	<pre>if(a==0) { break; }</pre>	==	<pre>if(a==0) { break; }</pre>	==	<pre>if(a==0) break;</pre>
----------------------------	----	--------------------------------	----	--------------------------------	----	----------------------------

즉 실행할 문장이 하나뿐이라면 블록을 사용해도 되고 사용하지 않아도 된다.
또한 줄바꿈을 해도 되고 한 줄에 써도 상관없다

03 기타제어문 : break문

```
1 //실습 7-9 break문 사용 예3
2 public class Ex07_09 {
3     public static void main(String[] args) {
4         int hap = 0;
5         int i;
6
7         for (i = 1; i <= 100; i++) {
8             hap = hap + i;           i값을 hap에 누적한다.
9
10            if (hap >= 1000)          hap이 1000보다 크거나 같으면
11                break;              for반복문을 빠져나간다.
12        }
13
14        System.out.printf(" 1~100의 합에서 최초로 1000이 넘는 위치는? : %d\n", i);
15    }
16 }
```

i값을 1부터 100까지 100회 실행한다.



- 10행을 보면 합계(hap)가 1000보다 크거나 같을 때 break문을 수행하여 루프를 빠져나감
- 현재 i값을 누적해가면서 1000을 넘는 순간의 i값을 구함. 즉, 결과는 1부터 44까지 더하면 1000미만이고, 45를 더하는 순간 1000이 넘는다는 의미임

03 기타제어문 : continue문

■ 반복문으로 다시 돌아가는 continue 문

- continue 문은 반복문인 for문, while문, do-while문에서만 사용
- 블록내부에서 continue문이 실행되면 for문의 증감식 또는 while문, do-while문의 조건식으로 이동함.
- continue 문을 만나면 블록의 남은 부분을 건너뛰고 반복문의 처음으로 돌아감

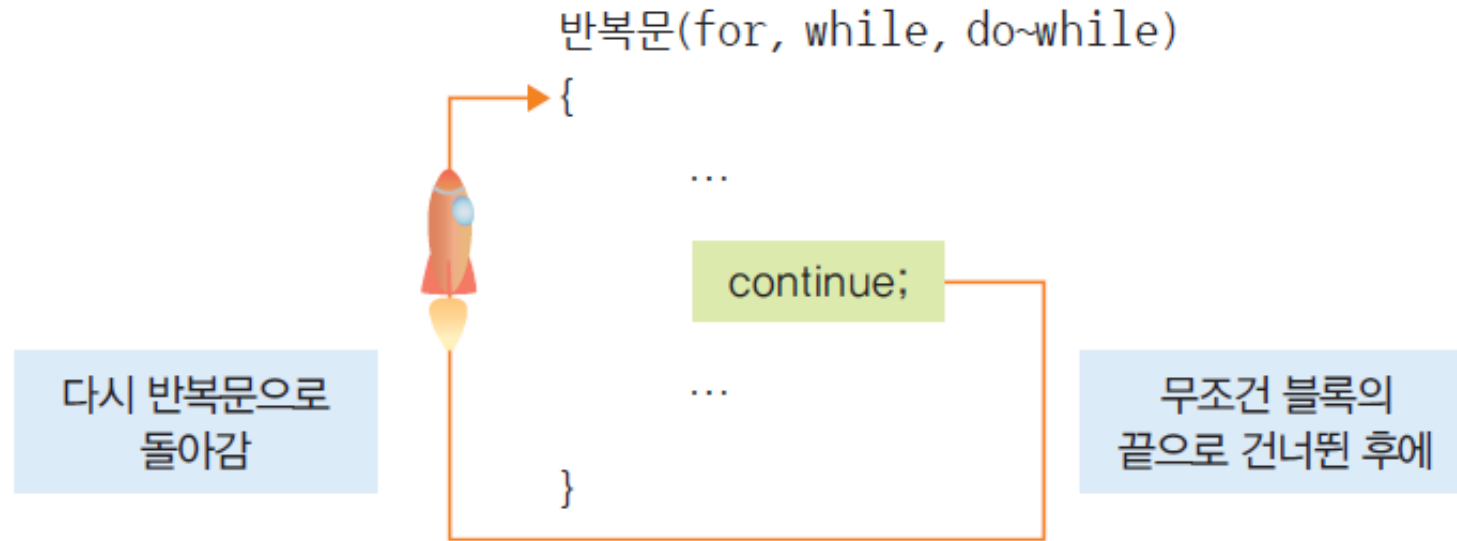
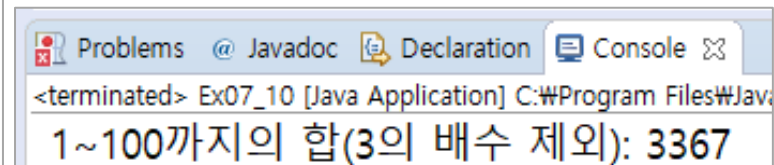


그림 7-15 continue 문의 작동

03 기타제어문 : continue문

```
1 //실습 7-10 continue문 사용 예
2 public class Ex07_10 {
3     public static void main(String[] args) {
4         int hap = 0;
5         int i;
6
7         for (i = 1; i <= 100; i++) {
8             if (i % 3 == 0)
9                 continue;
10
11             hap += i;
12         }
13
14         System.out.printf(" 1~100까지의 합(3의 배수 제외): %d\n", hap);
15     }
16 }
```



The screenshot shows a Java IDE window with tabs for Problems, Javadoc, Declaration, and Console. The Console tab is active, displaying the output of the program: "1~100까지의 합(3의 배수 제외): 3367". The window title is "<terminated> Ex07_10 [Java Application] C:\Program Files\Java\".

7행~11행 : 1부터 100까지 100회 반복

8행 ~9행 : i값을 3으로 나눈 나머지값이 0이면(3의배수) 블록의 끝으로 건너뛰고 다시 7행으로 돌아감

11행 : 3의 배수가 아닌 i값을 누적함

14행 : 누적된 값을 출력

이렇게 계속 진행하면 $hap=1+2+4+5+7+...$ 과 같은 계산식이 됨

03 기타제어문 : break 레이블문

■ 다중 반복문의 지정된 위치로 이동하는 break 레이블문

```
1 //실습 7-11 다중 반복문이 무한루프
2 public class Ex07_11 {
3     public static void main(String[] args) {
4         int hap = 0;
5         int i;
6
7         for (;;) {
8             for (i = 1; i <= 100; i++) {
9                 hap += i;
10                if (hap > 2000) {
11                    System.out.printf("%d\n", hap);
12                    hap = 0;
13                    break ;
14                }
15            }
16            System.out.printf("아직도 반복중...\n");
17        }
18    }
19 }
```

7행~16행 : 무한반복

8행~15행 : 100회 반복

9행 : 합계를 누적한다

10행~14행 : 누적된 값이 2000을 넘으면
hap을 출력하고 초기화한 다음 13행의 break로
반복문을 빠져나감

16행 : 무한반복을 위한 출력

이 코드는 무한 반복. 2000이 넘는 값을 만나면 13행에서
break를 만나 반복문을 빠져나가려고 시도.
하지만 가장 가까운 for 문(8~15행)의 끝인 15행을 빠져나가서
16행을 출력하고, 다시 6~16행의 무한 루프 for 문을 만나
8행부터 다시 시작.
결국 무한 반복.



```
Problems @ Javadoc Declared
<terminated> Ex07_11 [Java Application]

2016
아직도 반복중...
2016
아직도 반복중...
2016
아직도 반복중...
2016
아직도 반복중...
```

03 기타제어문 : break 레이블문

■ 다중 반복문의 지정된 위치로 이동하는 break 레이블문

- break 문을 별도로 지정한 레이블(label)과 함께 사용. 'break 레이블이름'과 같이 지정해줌

레이블 : 반복문(for, while, do~while)

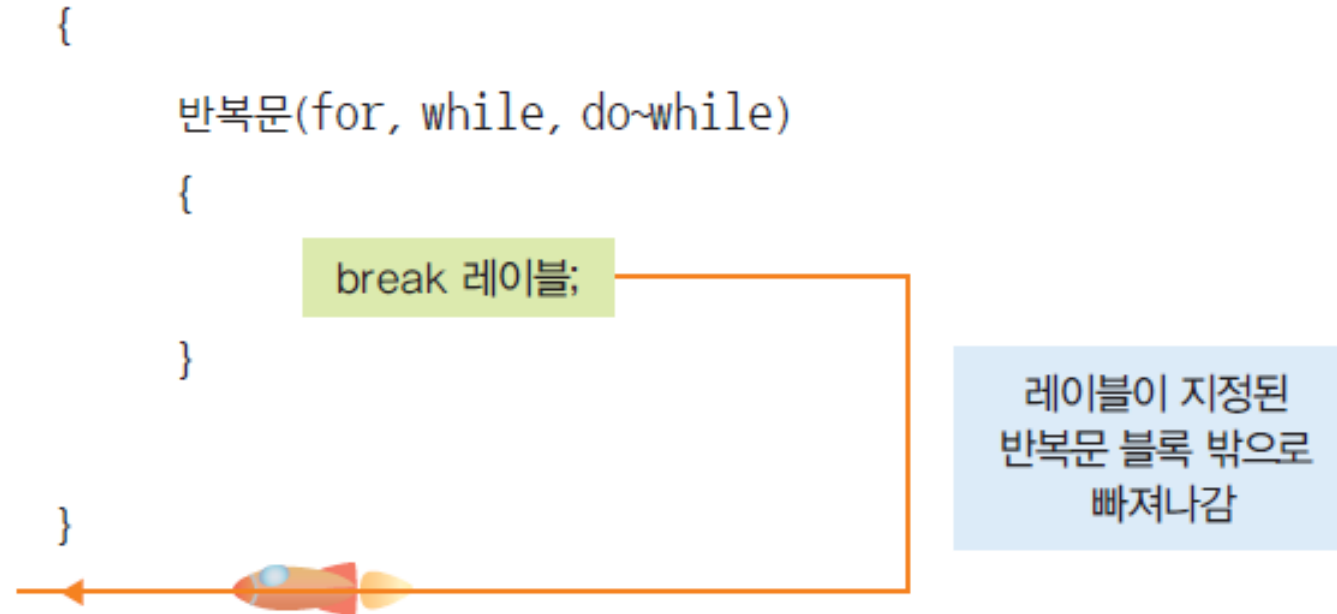
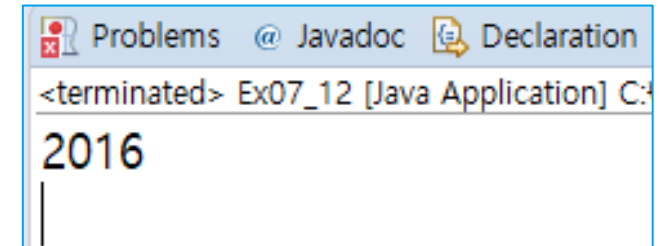


그림 7-18 break 레이블문의 작동

03 기타제어문 : break 레이블문

■ 다중 반복문의 지정된 위치로 이동하는 break 레이블문

```
1 //실습7-12 break레이블문 사용 예
2 public class Ex07_12 {
3     public static void main(String[] args) {
4         int hap = 0;
5         int i;
6
7         myLabel: for (;;) {           // 바깥 for문에 레이블을 지정한다
8             for (i = 1; i <= 100; i++) {
9                 hap += i;
10                if (hap > 2000) {
11                    System.out.printf("%d\\n", hap);
12                    hap = 0;
13                    break myLabel;      //지정된 mylabel의 반복문을 빠져나간다
14                }
15            }
16            System.out.printf("아직도 반복중...\\n");
17        }
18    }
19 }
```



13행에서 지정된 레이블의 반복문인 7~17행을 빠져나가므로 프로그램이 종료됨

03 기타제어문 : return문

■ 현재 메소드를 불렀던 곳으로 돌아가는 return 문

- return문은 현재 실행하고 있는 메소드를 끝내고, 메소드를 호출한 곳으로 돌아가게 하는 제어문임.
- return문을 만나면 프로그램은 종료되는 것은 아는데 현재 메소드인 main()을 빠져나가는 것이므로 프로그램이 종료되는 효과가 나타남

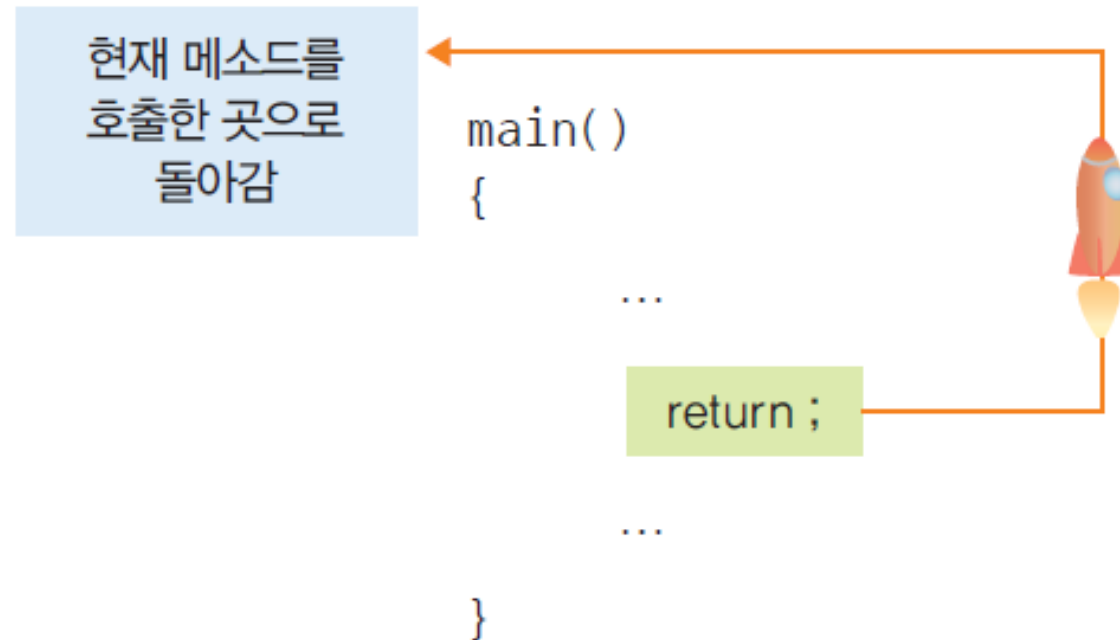


그림 7-20 return 문의 작동

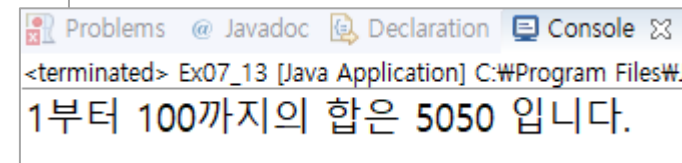
03 기타제어문 : return문

■ 현재 메소드를 불렀던 곳으로 돌아가는 return 문

- 예. 1부터 100까지 합계를 구하되 계산 중간에 합이 5000이 넘을 경우 메소드를 호출한 곳으로 돌아가는 프로그램을 작성해보자

```
1 //실습7-13 return문 사용 예
2 public class Ex07_13 {
3     public static void main(String[] args) {
4         int hap = 0;
5         int i;
6
7         for (i = 1; i <= 100; i++) //1부터 100까지 합계 누적
8             hap += i;
9
10        System.out.printf("1부터 100까지의 합은 %d 입니다.\n", hap); //합계 출력
11
12        if (hap > 5000)
13            return; //현재 메소드를 호출한 곳으로 복귀
14
15        System.out.printf("프로그램의 끝입니다."); //한번도 실행되지 않는다.
16    }
17 }
```

10행에서 누적된 합계를 출력한 다음 13행의 return문을 만나면 현재 메소드인 main()을 빠져나감.
15행은 한번도 실행되지 않음



03 기타제어문 : return문

■ return 값

- return 뒤에 아무것도 붙이지 않은 것은 현재 메소드인 main()의 형식이 void로 지정되었기 때문이다. void는 아무것도 없다는 의미이다.
- return 뒤에 붙이는 값은 현재 메소드의 데이터형(이 경우에는 void형)과 일치해야 한다.
- 만약 main 메소드가 int main()과 같이 되어 있다면 'return 0'과 같이 정수형 값을 써야 한다.

Self study 9-2

1. [실습7-9]에서 사용한 for문을 while 문으로 바꿔보자.
2. 다음 각 명령문에 대한 설명을 작성해보자.

break	continue	break레이블	return
-------	----------	----------	--------

질문은 이메일을 이용해주세요.
ds.june2@gmail.com

감사합니다