Kumoh National Institute of Technology

C Programming

Ch.3-1 반복 실행을 명령하는 반복문

컴퓨터공학과

C프로그래밍

Contents

1. while 문에 의한 문장의 반복
2. do ~ while 문에 의한 문장의 반복
3. for 문에 의한 문장의 반복
4. Visual C++에서 들여쓰기 옵션 (tip)

- tip : Visual C++ 사용 방법 등 참고 자료

퀴즈 : 1주차

[문제3] 정수 1개를 입력 받고, ×1에서 ×3까지 구구단의 일부를 출력하라.

- 입력 값이 3 이면, 3×1=3, 3×2=6, 3×3=9를 서로 다른 줄에 출력한다.

```c
#include <stdio.h>
int main(void)
{
    int i;
    scanf("%d", &i);
    printf("%d x %d = %d\n", i, 1, i * 1);
    printf("%d x %d = %d\n", i, 2, i * 2);
    printf("%d x %d = %d\n", i, 3, i * 3);
    return 0;
}
```

```
3
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
```

x1에서 x20까지 출력하려면?
x10에서 x20까지 출력하려면?
매 실행마다 출력 수가 달라지면?
(xa에서 xb까지 출력)

반복문이란?

- 반복문의 기능
  - 특정 영역을 특정 조건이 만족하는 동안 반복적으로 실행하기 위한 제어문
    - 예1) a 값이 10보다 작은 동안 a 값을 화면에 출력하라.
    - 예2) 사용자가 0을 입력할 까지 입력값을 모두 더하라.
- 세 가지 형태의 반복문
  - while 문
  - do ~ while 문
  - for 문

기본적인 형태는 다르나,
근본은 같다.

while 문의 기본 원리와 의미 (1)

```
int main(void)
{
    int num = 0;

    while (num < 5)
    {
        printf("Hello world! %d \n", num);
        num++;
    }
}
```

```
while( 반복 조건 )
{
    반복 내용
}
```

"반복 조건"이 참인 동안 "반복 내용"을 반복 실행하라!

```
Hello world! 0
Hello world! 1
Hello world! 2
Hello world! 3
Hello world! 4
```

일단 문법만 주목!
실행 결과 분석은 다음 페이지에서.

while 문의 기본 원리와 의미 (2)

- 반복 대상이 한 문장이면 중괄호 생략 가능

```
while(num<5)
    printf("Hello world! %d \n", num++);
```

```
while(num<5)
    printf("Hello world! %d \n", num), num++;
```

while 문의 기본 원리와 의미 (3)

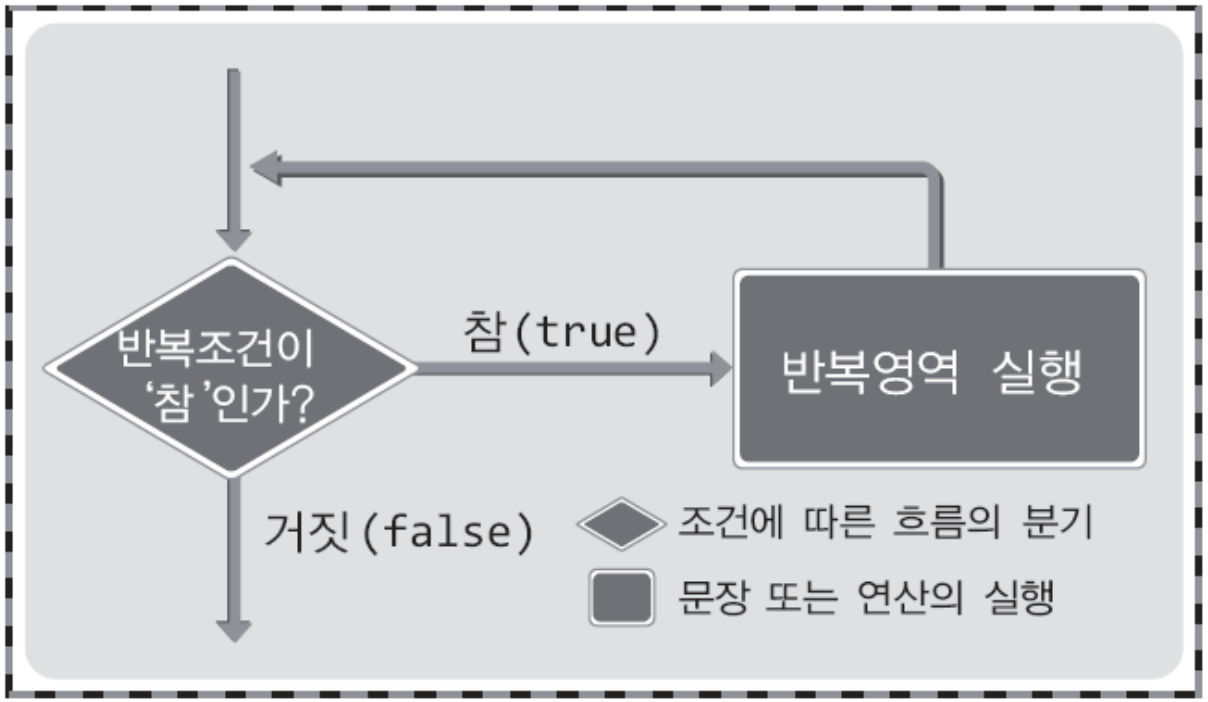
```
int main(void)
{
    int num = 0;

    while (num < 5)
    {
        printf("Hello world! %d \n", num);
        num++;
    }
}
```

변수 num의 값이 어떻게 변하는지, 출력 결과는 어떻게 되는지 분석해 보자.

```
Hello world! 0
Hello world! 1
Hello world! 2
Hello world! 3
Hello world! 4
```

while 문의 순서도

- 순서도 (flow chart)
  - 프로그램의 실행 흐름을 기호를 사용하여 그림으로 표현
- while 문의 순서도



들여쓰기를 생활화하자!

indentation!

- C 언어 : 다음줄, 들여쓰기가 의미에 영향을 미지치 않음
- 다음 두 코드를 비교해 보자.

```
int main(void)
{
    int num = 0;

    while (num < 5)
    {
        printf("Hello world! %d \r
        num++;
    }
}
```

```
int main(void)
{
int num = 0;
while (num < 5)
{
printf("Hello world! %d \n", num);
num++;
}
}
```

이해하기 어렵다

코드는 읽기 쉽게 작성하는 것이 중요!
- 공백줄, 들여쓰기 적절하게 사용
- 주석 사용

"4. Visual C++에서의 들여쓰기 옵션" 참고

while 문의 사용 예 : 구구단 출력

- num 변수의 초기값 및 변화에 주의

```
int main(void)
{
    int dan = 0, num = 1;

    printf("몇 단? ");
    scanf("%d", &dan);

    while (num < 10)
    {
        printf("%d × %d = %d \n", dan, num, dan * num);
        num++;
    }
}
```

```
몇 단? 7
7 × 1 = 7
7 × 2 = 14
7 × 3 = 21
7 × 4 = 28
7 × 5 = 35
7 × 6 = 42
7 × 7 = 49
7 × 8 = 56
7 × 9 = 63
```

Tip) 그래픽 문자 입력 방법
- 한글 자음 입력 → 한자키 클릭 → 윈도우의 오른쪽 하단에 그래픽 문자들 중 선택
- 곱셈 문자 : ㄷ 입력 → 한자키 → 선택
- Win + . 를 사용해도 됨

무한 루프가 발생되지 않도록 조심

- 무한 루프 : 무한히 반복해서 수행되는 상황

```
int main(void)
{
    int num = 0;

    while (num < 10)
    {
        printf("Hello, Mr. While \n");
    }
}
```

```
int main(void)
{
    int num = 0;

    while (1)
    {
        printf("Hello, Mr. While \n");
        num++;
    }
}
```

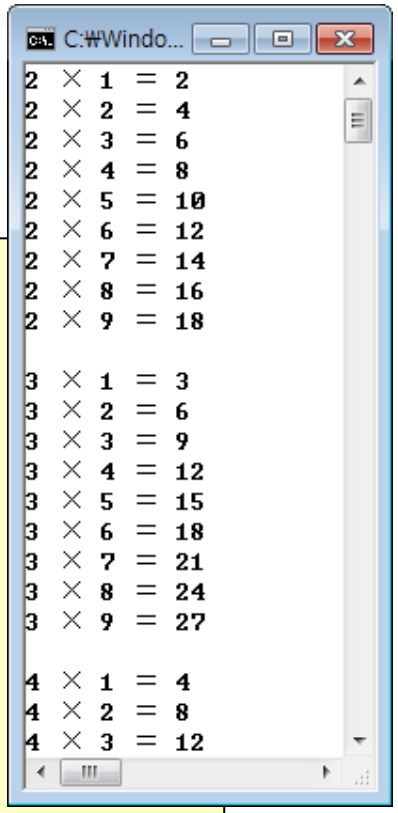
일반적으로는 무한 루프가 발생되면 안됨 → 주의
break 문 참고(다음 ppt 자료)

while 문의 중첩

- while 문 안에 while 문이 중첩해서 나올 수 있음
- 구구단 출력

```
int main(void)
{
    int cur = 2;
    int is = 0;

    while (cur < 10)      // 2단부터 9단까지 반복
    {
        is = 1;           // 새로운 단의 시작을 위해서
        while (is < 10)   // 각 단의 1부터 9의 곱을 표현
        {
            printf("%d × %d = %d \n", cur, is, cur * is);
            is++;
        }
        cur++;    // 다음 단으로 넘어가기 위한 증가
        printf("\n");
    }
}
```

실행 흐름에 주의



do ~ while 문의 기본 구성

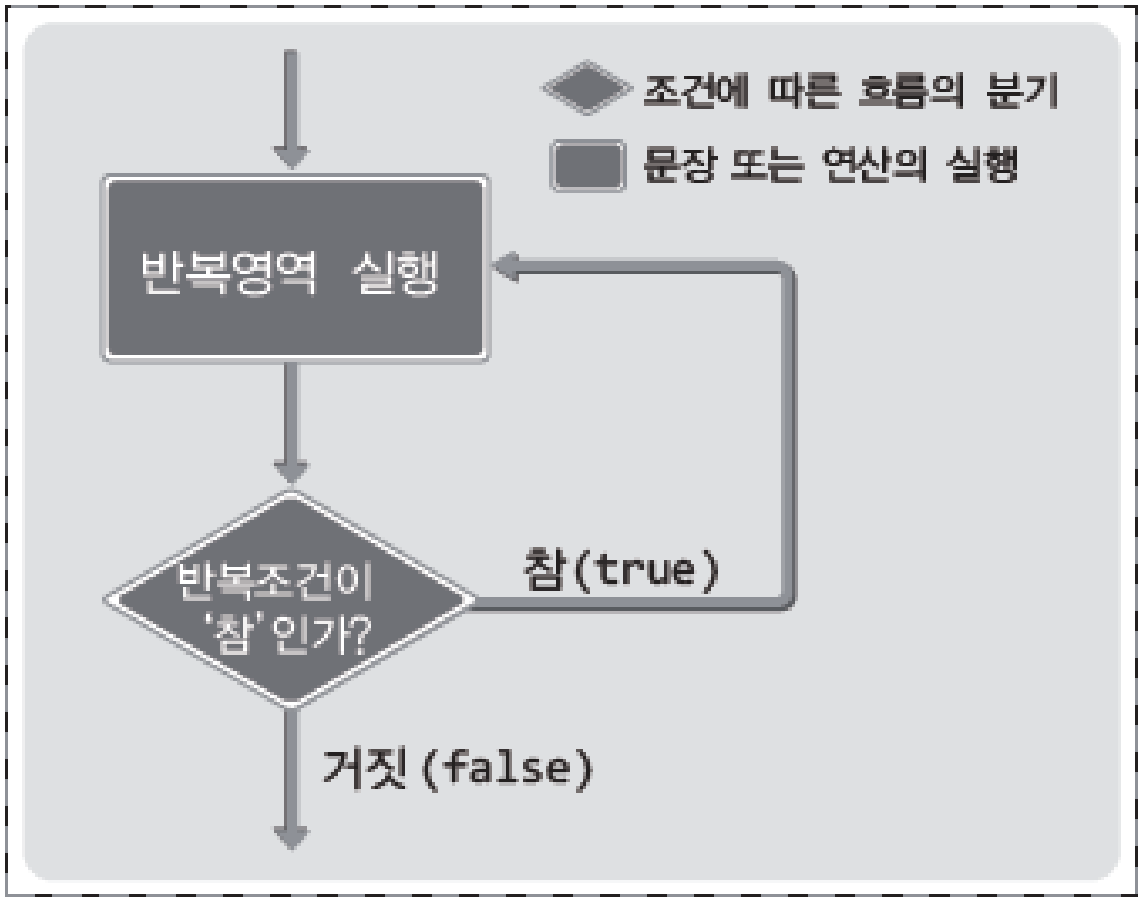
- while (조건)의 조건이 참인 동안 반복 실행

```
do
{
    printf("Hello world! \n");
    num++;
} while(num<3);
```

실행 흐름을 분석해 보자.
num = 0; 가정

- while 문과의 차이점
  - do ~ while 문은 일단 한 번 실행한 후 조건 검사 수행
    - do ~ while 문은 반드시 한 번은 실행됨

do ~ while 문의 순서도



do ~ while 문의 사용 예

- 사용자가 입력한 정수들의 합 구하기. 단, 0을 입력하면 종료

```
int main(void)
{
    int total = 0, num = 0;

    do
    {
        printf("정수 입력(0 to quit): ");
        scanf("%d", &num);
        total += num;
    } while (num != 0);

    printf("합계: %d \n", total);
}
```

최소한 한 번은 실행되어야 하므로 do ~ while문이 자연스러움
- while 문으로도 구현 가능 → while 문을 사용해서 작성해 보라.

```
정수 입력(0 to quit): 1
정수 입력(0 to quit): 2
정수 입력(0 to quit): 3
정수 입력(0 to quit): 4
정수 입력(0 to quit): 5
정수 입력(0 to quit): 0
합계: 15
```

반복문의 필수 3요소

- while 문으로 본 일반적인 상황에서의 반복문의 3요소
  - 정해진 횟수를 반복하기 위한 하나의 변수
  - 계속 수행할 것인지 검사하는 조건 검사
  - 변수의 값을 변경하기 위한 연산

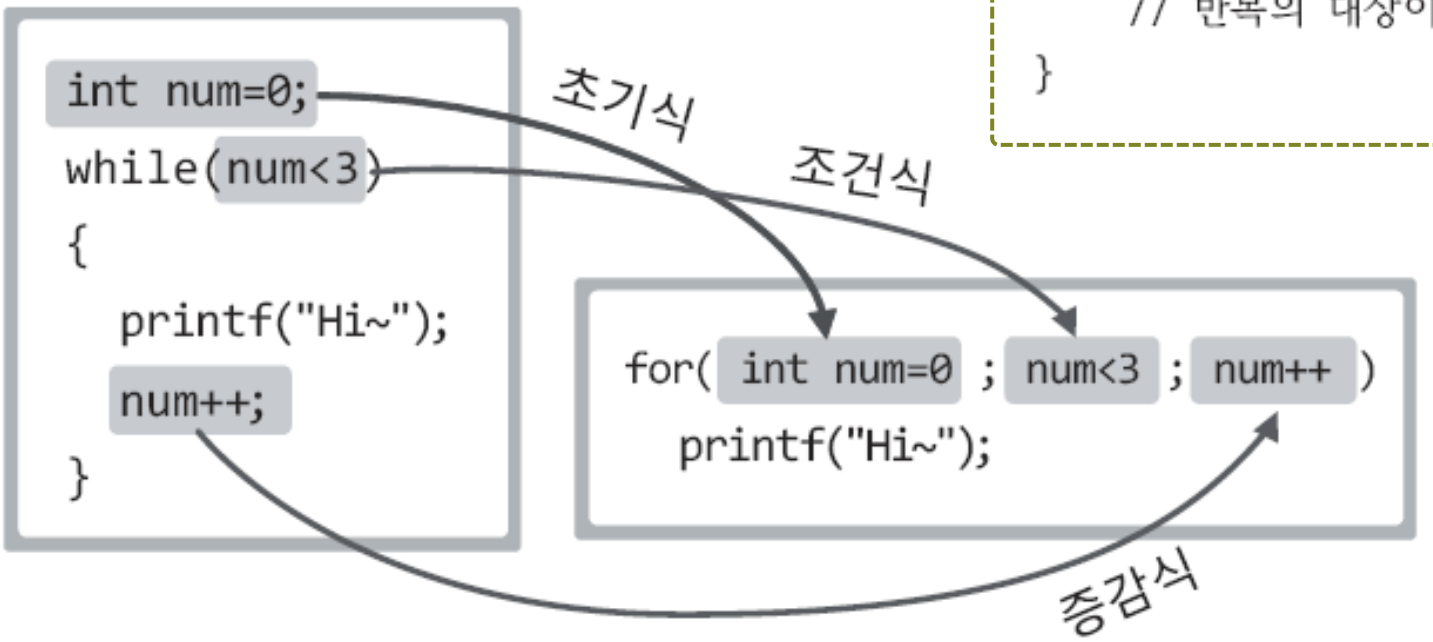
```
int main(void)
{
    int num=0;    // 필수요소 1. 반복을 위한 변수의 선언
    while(num<3)    // 필수요소 2. 반복의 조건검사
    {
        printf("Hi~");
        num++;    // 필수요소 3. 반복의 조건을 '거짓'으로 만들기 위한 연산
    }
    . . . .
}
```

- 3요소를 한 줄에 표시할 수 있는 반복문 → for 문

for 문의 구조와 이해

- 초기식, 조건식, 증감식 모두 포함
- while 문과의 비교

```
for( 초기식 ; 조건식 ; 증감식 )
{
    // 반복의 대상이 되는 문장들
}
```



- int num;은 for 문 앞에서 선언될 수도 있음
- 반복 문장이 1개이면 중괄호 생략 가능

for 문의 흐름 이해

for문의 구성요소

√ 초기식 본격적으로 반복을 시작하기에 앞서 딱 한번 실행된다.

√ 조건식 매 반복의 시작에 앞서 실행되며, 그 결과를 기반으로 반복 여부 결정!

√ 증감식 매 반복 실행 후 마지막에 연산이 이뤄진다.

- 첫 번째 반복의 흐름
  1 → 2 → 3 → 4 [num=1]
- 두 번째 반복의 흐름
  2 → 3 → 4 [num=2]
- 세 번째 반복의 흐름
  2 → 3 → 4 [num=3]
- 네 번째 반복의 흐름
  2 [num=3] 따라서 탈출!

```
    1          2        4
for( int num=0 ; num<3 ; num++ )
{  3
   printf("Hi~");
}
```

for 문의 다양한 사용 예 (1)

```
int main(void)
{
    int total = 0;
    int i, num;

    printf("0부터 num까지의 덧셈, num은? ");
    scanf("%d", &num);

    for (i = 0; i < num + 1; i++)
        total += i;

    printf("0부터 %d까지 덧셈결과: %d \n", num, total);
}
```

- 다음과 비교해 보라.
for (i = 0; i <= num; i++)

```
0부터 num까지의 덧셈, num은? 10
0부터 10까지 덧셈결과: 55
```

for 문의 다양한 사용 예 (2)

```
int main(void)
{
    double total = 0.0;
    double input = 0.0;
    int num = 0;

    for ( ; input >= 0.0; )
    {
        total += input;
        printf("실수 입력(minus to quit) : ");
        scanf("%lf", &input);
        num++;
    }

    printf("평균: %f \n", total / (num - 1));
}
```

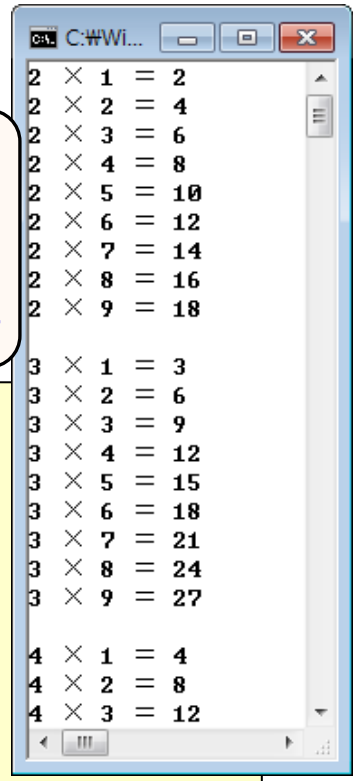
```
실수 입력(minus to quit) : 3.2323
실수 입력(minus to quit) : 5.1891
실수 입력(minus to quit) : 2.9297
실수 입력(minus to quit) : -1.0
평균: 3.783700
```

- 초기식, 조건식, 증감식 생략 가능
- 초기식, 증감식 생략 : while 문과 동일
- 조건식 생략 : 항상 참으로 인식
  → 무한 루프로 동작 [조심!]

for 문의 다양한 사용 예 (3)

- for 문의 중첩 사용 : 구구단

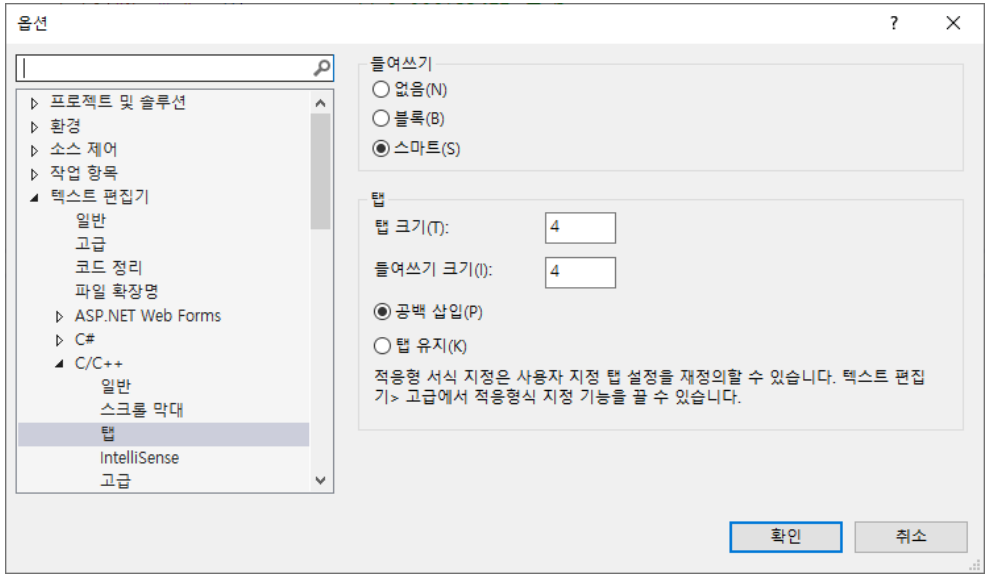
- while, do ~ while, for 문 모두 중첩 사용 가능
  예) while 문 안에 for 문 사용
- 중첩문 내에서 또 다시 중첩문 사용 가능
  예) while 문 안에 for 문 사용, 그 for 문 안에 또 while 문 사용




```
int main(void)
{
    int cur, is;

    for (cur = 2; cur < 10; cur++)
    {
        for (is = 1; is < 10; is++)
            printf("%d × %d = %d \n", cur, is, cur * is);

        printf("\n");
    }
}
```

VC++에서 들여쓰기 옵션 변경 방법

- 탭(Tab) 키 클릭 시 기본 동작 방식
  - 탭(\t) 문자 입력
  - 문제점 : 소스 코드를 다른 프로그램으로 복사 시 이상하게 보일 수 있음 → 프로그램마다 탭을 처리하는 방식 다름
- 탭 키 클릭 시 공백 문자 4칸 삽입 방법
  - 메뉴 [도구]-[옵션]-[텍스트 편집기]-[C/C++][탭] : 공백 삽입 선택
    - 탭 크기 : 4, 들여쓰기 : 4
  - 들여쓰기 : Tab 키
  - 내어쓰기 : Shift-Tab 키
- 기존 소스 코드의 들여쓰기 적용 방법
  - 소스 선택 후 Alt+F8 키



이번 장에서 배운 것

- 반복문은 어떤 조건을 만족하는 동안 문장(들)을 반복적으로 실행하고자 할 때 사용하는 제어문이다.
- 반복문에는 while 문, do ~ while 문, for 문이 있다.
- while 문은 조건을 검사하여 조건이 참이면 실행하게 된다.
- do ~ while 문은 먼저 문장(들)을 실행한 후 조건을 검사한다. 1회 이상 실행한다면 do ~ while 문이 더 적합할 수도 있다.
- for 문은 초기식, 조건식, 증감식을 모두 포함한다.
- 반복문 내에 또 다시 반복문을 사용할 수 있다.
- 반복문 사용 시 의도한 것이 아니라면 무한 루프가 발생되지 않도록 조심해야 한다.