Kumoh National Institute of Technology

C Programming

Ch.3-2 조건에 따른 흐름의 분기

컴퓨터공학과

C프로그래밍

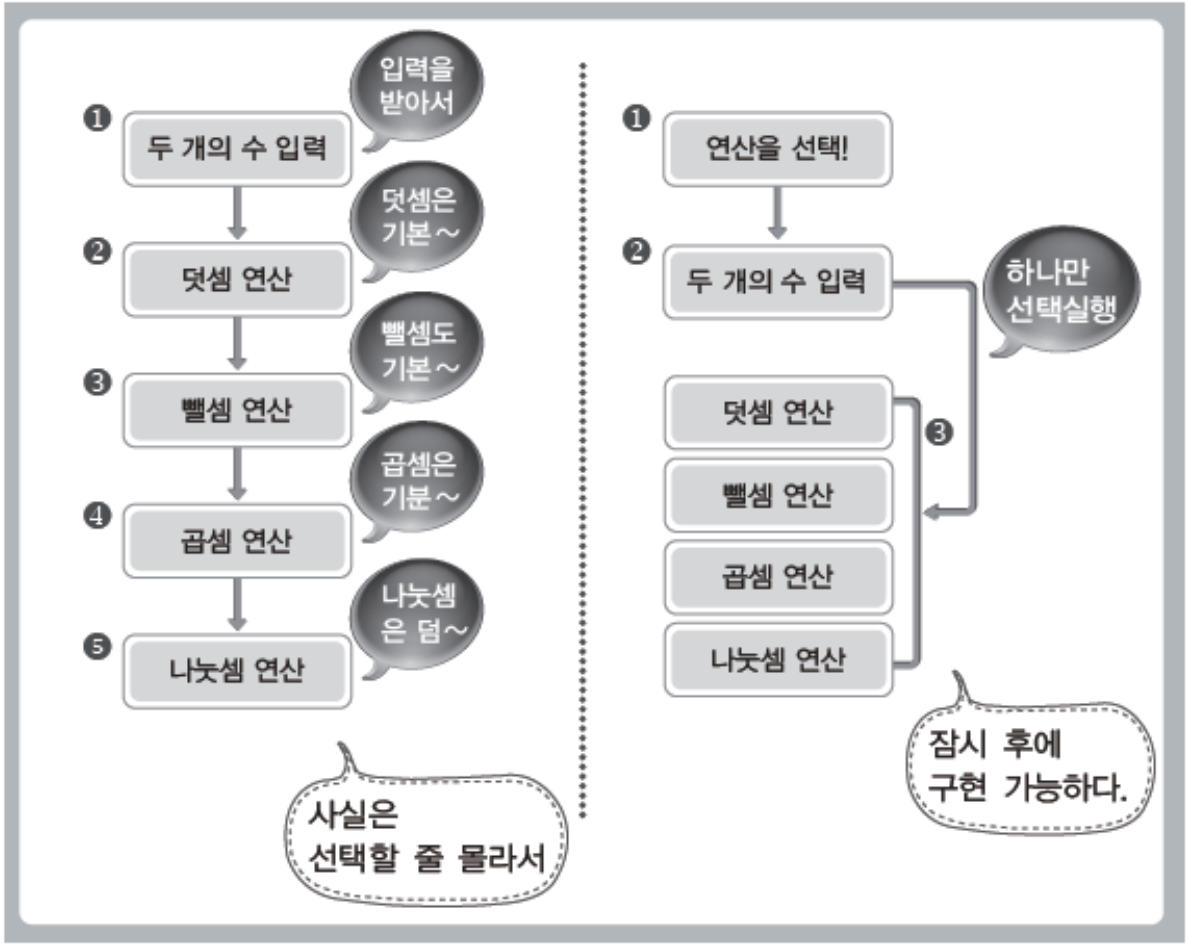
Contents

1. 조건적 실행과 흐름의 분기 : if 문
2. 반복문의 생략과 탈출 : continue & break
3. switch 문에 의한 선택적 실행
4. goto 문

흐름의 분기가 필요한 이유

- 상황에 따른 프로그램의 유연성 부여

사용자의 입력에 따라 사칙연산 중 하나만 선택하여 실행하고자 할 때
→ 조건을 따지는 조건문 필요



if 문을 이용한 조건적 실행 (1)

- 조건(expr)의 평가 결과가 참이면(="0이 아닌 값") statement를 실행하고 거짓이면(="0") 실행하지 않음

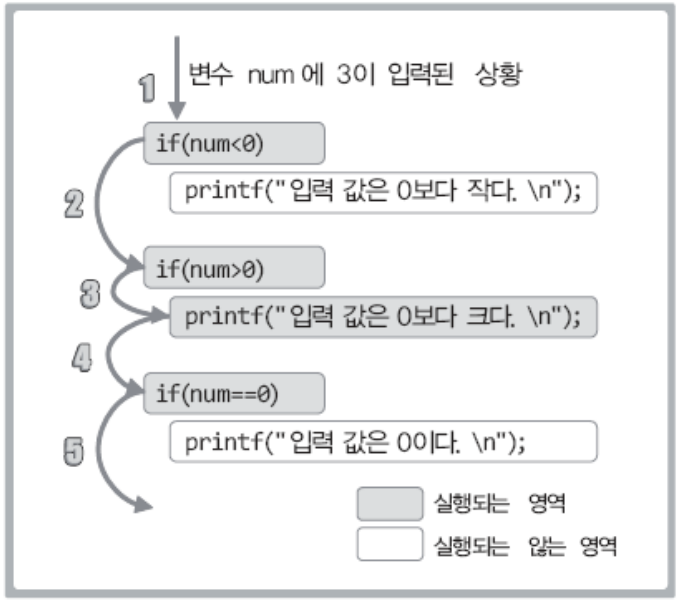
```
if(num1>num2)   num1이 num2보다 크면 실행
{
    printf("num1이 num2보다 큽니다. \n");
    printf("%d > %d \n", num1, num2);
}
```

```
if(num1>num2)   한 줄이면 중괄호 생략 가능
    printf("num1이 num2보다 큽니다. \n");
```

```
// 문법
if (expr)
    statement

if (expr)
{
    statement1;
    statement2;
    ......
    statementn;
}
```

if 문을 이용한 조건적 실행




```
int main(void)
{
    int num;
    printf("정수 입력: ");
    scanf("%d", &num);

    if (num < 0)      // num이 0보다 작으면 아래의 문장 실행
        printf("입력 값은 0보다 작다. \n");

    if (num > 0)      // num이 0보다 크면 아래의 문장 실행
        printf("입력 값은 0보다 크다. \n");

    if (num == 0)      // num이 0이면 아래의 문장 실행
        printf("입력 값은 0이다. \n");
}
```

```
정수 입력: 3
입력 값은 0보다 크다.
```

```
정수 입력: 0
입력 값은 0이다.
```

예제 : if 문을 이용한 계산기 프로그램

```
int main(void)
{
    int opt;
    double num1, num2;
    double result;

    printf("1.덧셈 2.뺄셈 3.곱셈 4.나눗셈 \n");
    printf("선택? ");
    scanf("%d", &opt);
    printf("두 개의 실수 입력: ");
    scanf("%lf %lf", &num1, &num2);

    if (opt == 1)
        result = num1 + num2;
    if (opt == 2)
        result = num1 - num2;
    if (opt == 3)
        result = num1 * num2;
    if (opt == 4)
        result = num1 / num2;

    printf("결과: %f \n", result);
}
```

사용자 입력에 따라 사칙연산 중 하나만 실행.
그러나 4번의 조건 모두 검사
→ if ~ else 문으로 해결 (7페이지)

```
1.덧셈 2.뺄셈 3.곱셈 4.나눗셈
선택? 3
두 개의 실수 입력: 2.14 5.12
결과: 10.956800
```

예제 : 3의 배수이거나 4의 배수인 정수 출력

- 1 이상 100 미만의 정수 중에서 3의 배수 또는 4의 배수만 출력

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

    for (num = 1; num < 100; num++)
    {
        if (num % 3 == 0 || num % 4 == 0)
            printf("3 또는 4의 배수: %d \n", num);
    }
}
```

```
3 또는 4의 배수: 3
3 또는 4의 배수: 4
3 또는 4의 배수: 6
3 또는 4의 배수: 8
3 또는 4의 배수: 9
3 또는 4의 배수: 12
......
```

특정 범위 내의 값들에 대한 처리
→ 반복문들 중 for 문이 적절

if ~ else 문을 이용한 흐름의 분기

- 조건(expr)의 평가 결과가 참이면 문장1을 실행하고, 거짓이면 문장2를 실행함

```
// 문법
if (조건)
    문장1;
else
    문장2;

if (조건)
{
    문장1-1;
    ......
    문장1-n;
}
else
{
    문장2-1;
    ......
    문장2-m;
}
```

```
int main(void)
{
    int num;
    printf("정수 입력: ");
    scanf("%d", &num);

    if (num < 0)
        printf("입력 값은 0보다 작다. \n");
    else
        printf("입력 값은 0보다 작지 않다. \n");
}
```

```
정수 입력: 7
입력 값은 0보다 작지 않다.
```

if ~ else if ~ else 문의 구성

- if ~ else if 문 : 만족되는 조건을 만나면 해당 문장을 실행하고 만족되는 조건이 없으면 else 문 실행

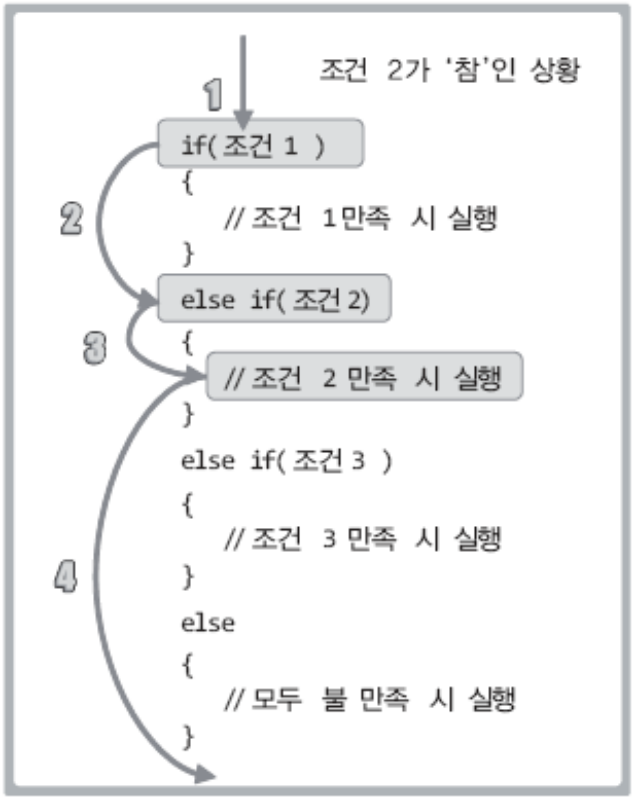
```
if( 조건 1 )
{
    // 조건 1 만족 시 실행
}
else if( 조건 2 )
{
    // 조건 2 만족 시 실행
}
else if( 조건 3 )
{
    // 조건 3 만족 시 실행
}
else
{
    // 모두 불 만족 시 실행
}
```

마지막 else는 생략 가능

얼마든지 추가 삽입 가능!

```
else if( 조건 4 )
{
    // 조건 4 만족 시 실행
}
```

if...else if...else문의 구성



if...else if...else문의 흐름

if ~ else if ~ else 문의 사용 예

```
int main(void)
{
    int opt;
    double num1, num2;
    double result;

    printf("1.덧셈 2.뺄셈 3.곱셈 4.나눗셈 \n");
    printf("선택? ");
    scanf("%d", &opt);
    printf("두 개의 실수 입력: ");
    scanf("%lf %lf", &num1, &num2);

    if (opt == 1)
        result = num1 + num2;
    else if (opt == 2)
        result = num1 - num2;
    else if (opt == 3)
        result = num1 * num2;
    else
        result = num1 / num2;

    printf("결과: %f \n", result);
}
```

사칙연산 계산기 프로그램 보완

하나가 참이 되면 if ~ else 전체를 skip함

if ~ else if ~ else 문의 진실

```
if(num<0)
    printf("입력 값은 0보다 작다. \n");
else if(num>0)
    printf("입력 값은 0보다 크다. \n");
else
    printf("입력 값은 0이다. \n");
```

```
if(num<0)
{
    printf("입력 값은 0보다 작다. \n");
}
else
{
    if(num>0)
        printf("입력 값은 0보다 크다. \n");
    else
        printf("입력 값은 0이다. \n");
}
```

if~else문은 하나의 문장임을 상기!

```
if(num<0)
   printf("입력 값은 0보다 작다. \n");
else
      if(num>0)
   printf("입력 값은 0보다 크다. \n");
else
   printf("입력 값은 00이다. \n");
```

else에 하나의 if~else문이 속한 상황.
속한 문장이 하나일 때에는 중괄호를 생략할 수 있다!

중첩 if 문의 해석 주의 (참고)

- if (조건) ~ if (조건) ~ else ~ 문장의 해석?

```
if (a != b)
    if (a > b)
        printf("a가 크다.\n");
else
    printf("같다.\n");
```

```
if (a != b)
    if (a > b)
        printf("a가 크다.\n");
    else
        printf("b가 크다.\n");
```

마지막 else는 어떤 if와 짝이 될까?
두 번째가 맞음 : 원칙 – 가장 가까운 if와 대응됨
아래와 같이 의미가 명확하게 프로그램을 작성하도록!

```
if (a != b)
{
    if (a > b)
        printf("a가 크다.\n");
}
else
    printf("같다.\n");
```

```
if (a != b)
{
    if (a > b)
        printf("a가 크다.\n");
    else
        printf("b가 크다.\n");
}
```

조건 연산자 : 피연산자가 3개인 삼항 연산자

- 조건이 참이면 data1 반환, 거짓이면 data2 반환
  - if ~ else 문을 간결히 표현할 때 사용 가능

```
(num1>num2) ? (num1) : (num2);

(조건) ? data1 : data2
```

```
int main(void)
{
    int num, abs;
    printf("정수 입력: ");
    scanf("%d", &num);

    abs = num > 0 ? num : num * (-1);
    printf("절댓값: %d \n", abs);
}
```

```
정수 입력: -79
절댓값: 79
```

```
// 아래 코드와 동일
if (num > 0)
    abs = num;
else
    abs = num * (-1);
```

break : 이제 그만 반복문을 빠져나가자!

- 강제로 반복문을 빠져나가고자 할 때 break 사용
  - 해당 반복문에 한해 빠져나옴 (가장 가까운 반복문)

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

    while (1)
    {
        sum += num;
        if (sum>5000)
            break;
        num++;
    }

    printf("sum: %d \n", sum);
    printf("num: %d \n", num);
}
```

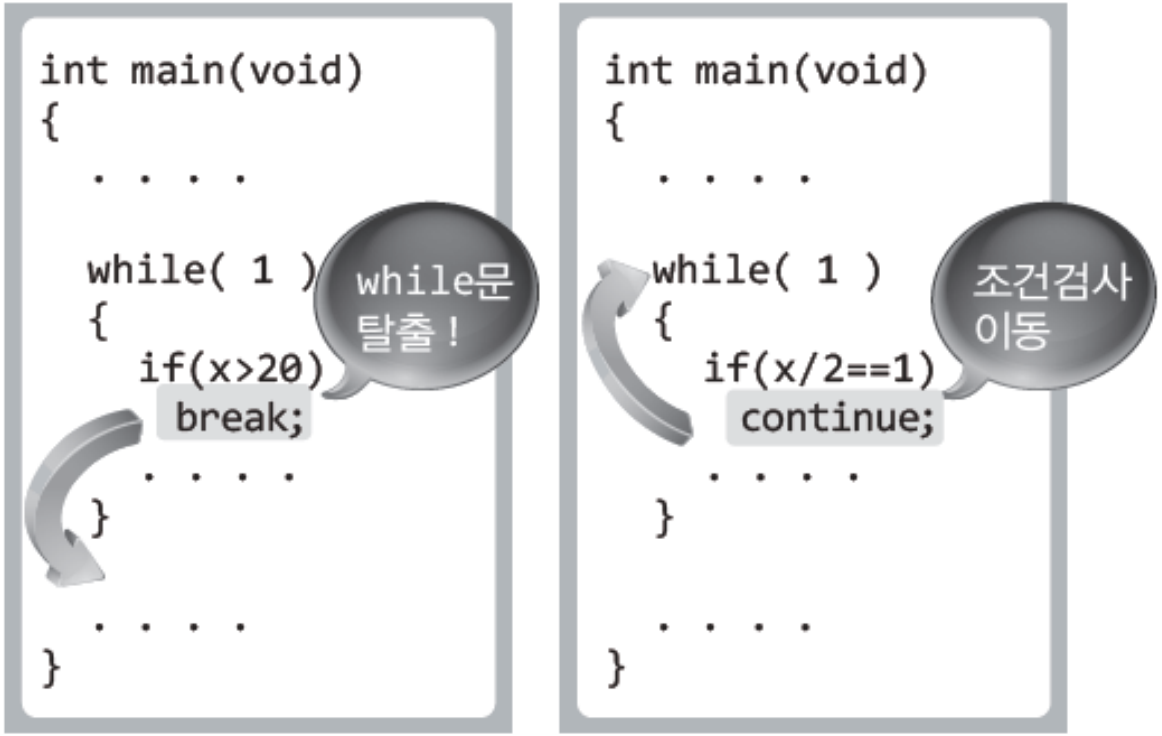
주로 if 문과 함께 사용하여 특정 조건이 만족될 때 반복문을 빠져나오는 용도로 사용

```
sum: 5050
num: 100
```

가장 가까운 반복문을 탈출한다!

continue : 나머지 문장(들)은 생략하고 반복문의 머릿줄로 가자

- 반복문에서 continue 이후의 문장들을 skip하고 다시 반복문의 머릿줄로 이동
- break와 continue 비교



continue 문은 반복문을 빠져나가지 않는다! 다시 반복조건을 확인하러 처음으로 이동

1. 반복문의 머릿줄로 이동한다.
2. continue이후의 문장을 생략한다(건너뛴다).

continue 문의 사용 예

```
int main(void)
{
    int num;
    printf("start! ");

    for (num = 0; num < 20; num++)
    {
        if (num % 2 == 0 || num % 3 == 0)
            continue;

        printf("%d ", num);
    }

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

2의 배수이거나 3의 배수이면 출력 생략

1. 반복문의 머릿줄로 이동한다.
2. continue이후의 문장을 생략한다(건너뛴다).

```
start! 1 5 7 11 13 17 19 end!
```

switch 문의 구성과 기본 기능

- switch, case, default, break 문으로 구성

- 여러 개의 "같다" 조건 검사를 수행
- if 문으로 변환 가능
- default 생략 가능 : if ~ else의 else 역할

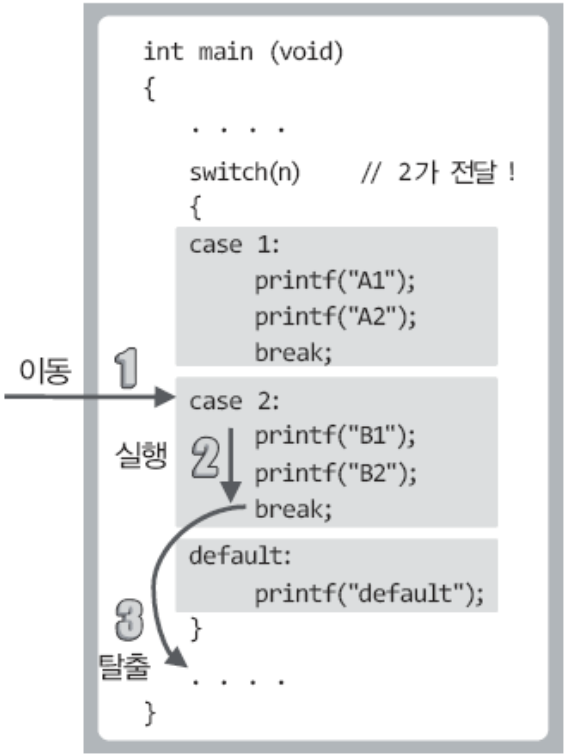
```
int main (void)
{
    . . . . .
    switch(n)
    {
    case 1:
        printf("A1");
        printf("A2");
        break;
    case 2:
        printf("B1");
        printf("B2");
        break;
    default:
        printf("default");
    }
    . . . . .
}
```

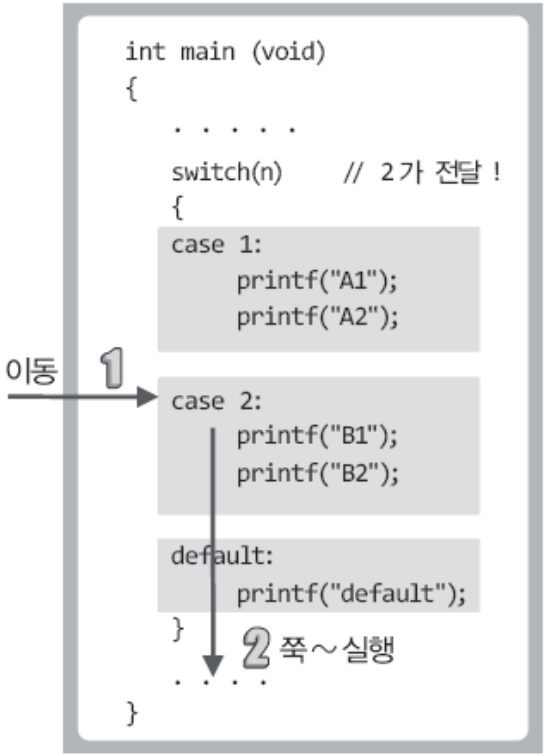
case 1 영역

case 2 영역

default 영역

switch문의 기본구성





삽입되어 있는 break문이 갖는 의미

switch 문 사용 예

```
int main(void)
{
    int num;
    printf("1 이상 4 이하의 정수 입력: ");
    scanf("%d", &num);

    switch (num)
    {
    case 1:
        printf("1은 ONE \n");
        break;
    case 2:
        printf("2는 TWO \n");
        break;
    case 3:
        printf("3은 THREE \n");
        break;
    case 4:
        printf("4는 FOUR \n");
        break;
    default:
        printf("I don't know! \n");
    }
}
```

- num 자리에 올 수 있는 자료형
  : 정수형(char, short, int, long int)
- switch 문은 “== 조건”만 검사
  : 작다, 크다 등의 비교는 불가능

```
1 이상 5 이하의 정수 입력: 3
3은 THREE
```

break 문을 생략한 예

```
int main(void)
{
    char sel;
    printf("M 오전, A 오후, E 저녁 \n");
    printf("입력: ");
    scanf("%c", &sel);

    switch (sel)
    {
    case 'M':
    case 'm':
        printf("Morning \n");
        break;
    case 'A':
    case 'a':
        printf("Afternoon \n");
        break;
    case 'E':
    case 'e':
        printf("Evening \n");
        break;     // 사실 불필요한 break문!
    }
}
```

한 번 참이 되면 계속 참으로 인식
- 'M' 또는 'm'이면 "Morning" 출력 후 빠져나감
- 다음과 같이 한 줄에 작성 가능
  case 'M' : case 'm' :

```
M 오전, A 오후, E 저녁
입력: M
Morning
```

switch vs. if ~ else if ~ else (1)

- 분기의 수가 많아지면 switch 문이 간결해 보임

```
if(n ==1)
   printf("AAA");
else if(n==2)
   printf("BBB");
else if(n==3)
   printf("CCC");
else
   printf("EEE");
```

VS.

```
switch(n)
{
 case1:
    printf("AAA");
    break;
 case2:
    printf("BBB");
    break;
 case3:
    printf("CCC");
    break;
 default:
    printf("EEE");
}
```

switch vs. if ~ else if ~ else (2)

- if ~ else의 표현 능력 > switch의 표현 능력
  - 모든 switch 문은 if ~ else 문으로 대체 가능
  - 역은 거짓 (될 수도 있고, 안 될 수도 있음)

```
if (0<=n && n<10)
    printf("0이상 10미만");

else if(10<=n && n<20)
    printf("10이상 20미만");

else if(20<=n && n<30)
    printf("20이상 30미만");

else
    printf("30이상 ");
```

```
switch(n)
{
 case ??? :
    printf("0이상 10미만");
    break;
 case ??? :
    printf("10이상 20미만");
    break;
 case ??? :
    printf("20이상 30미만");
    break;
 default:
    printf("30이상 ");
}
```

무조건 원하는 레이블로 이동하는 goto 문

```
int main(void)
{
    int num;
    printf("자연수 입력: ");
    scanf("%d", &num);

    if (num == 1)
        goto ONE;
    else if (num == 2)
        goto TWO;
    else
        goto OTHER;

ONE:
    printf("1을 입력하셨습니다! \n");
    goto END;
TWO:
    printf("2를 입력하셨습니다! \n");
    goto END;
OTHER:
    printf("3 혹은 다른 값을 입력하셨군요! \n");

END: ;
}
```

이해하기 힘든 코드가 될 수 있으므로 goto 문의 사용 자제

```
자연수 입력: 2
2를 입력하셨습니다!
```

레이블 (label)

- 레이블 다음에는 반드시 1개 이상의 문장이 나와야 됨
  ; → 빈 문장. 이것도 문장임

이번 장에서 배운 것

- 조건문은 어떤 조건에 따라 실행 흐름을 변경하고자 할 때 사용하는 제어문이다.
- 조건문에는 if 문과 switch 문이 있다.
- if 문은 if 문, if ~ else 문, if ~ else if ~ else 문으로 활용이 가능하다.
- switch 문은 여러 가지 "같다" 조건에 따른 문장들을 실행하고자 할 때 사용하며, case, default, break 문이 함께 사용된다.
- break는 반복문을 탈출할 때 사용하고, continue 문은 반복문 내의 이후의 문장들을 skip하고 반복문의 처음으로 이동할 때 사용한다.
- goto 문을 사용하여 특정 레이블로 이동할 수 있지만, 복잡한 코드가 만들어질 가능성이 높으므로 가급적 사용하지 않는다.