<4장> 조건문

01

제어 흐름

1. 제어 흐름

제어 흐름

- 프로그램이 실행되는 흐름을 표현하는 방법

순차

- 지금까지 main() 안에 있는 코드를 한 문장(statement)씩 순차적으로 (sequentially) 실행
- 중간에 printf() 같은 함수나 scanf() 함수를 실행하지만, 순차적 흐름의 한 문장

선택

- 주어진 조건에 따라 다른 코드를 실행시키는 것

반복

- 특정 작업을 거듭 수행하도록 하는 경우에 사용

02

조건 표현식

2. 조건 표현식

- 조건식 또는 조건 표현식(conditional expression)은 참/거짓의 불린(boolean) 값을 반환
- 조건식의 결과에 따라 선택적으로 후속 작업의 실행을 결정
- 조건식에는 상수, 변수, 연산식이 사용
- ANSI C는 불린 자료형이 따로 없음

표 4-1 참/거짓을 표현하는 표현식의 예

불린 값	표현식의 예
참	1, 2 또는 -1 -2처럼 0이 아닌 모든 정수 0.0이 아닌 모든 실수 NULL이 아닌 메모리 주소 '\0'이 아닌 모든 문자 결괏값이 0 또는 0.0이 아닌 연산식(예 : 2 + 3) 등
거짓	0, 0.0, 널('\0') 문자 NULL 포인터 결괏값이 0 또는 0.0인 연산식(예 : 2 - 2) 등

2. 조건 표현식

관계 연산자

- 두 피연산자를 비교해서 참/거짓에 해당하는 값을 생성

표 4-2 관계 연산자(A, B는 피연산자 두 개를 나타낸다)

연산자	설명
A < B	A가 B보다 작으면 T. 아니면 F
A > B	A가 B보다 크면 T. 아니면 F
A <= B	A가 B보다 작거나 같으면 T. 아니면 F
A >= B	A가 B보다 크거나 같으면 T. 아니면 F
A == B	A가 B와 같으면 T. 아니면 F
A != B	A가 B와 같지 않으면 T. 아니면 F

2. 조건 표현식

관계 연산자

- 주의 사항
  - <=, >=, != 는 순서에 맞게 사용
  - 주로 사용할 수 있는 자료형으로는 기본형(정수, 실수, 문자형), 포인터 등
  - 실수는 저장 방법 때문에 정확하게 비교되지 않을 수 있음
  - 대입 연산자 '='와 혼동하지 말아야 함

코드 4-1

```
1    int n = 3;
2    int boolean1 = (n == 2);
3    int boolean2 = (n = 2);
```

※ 여러 가지 관계 연산자를 사용하는 코드

➔ 코드 4-2
➔ 실행 결과

2. 조건 표현식

관계 연산자

enum 자료형 - 비교 연산

- 등급으로 나뉜 멤버십 프로그램

```
typedef enum { NONE, FRIENDS, SILVER, GOLD, PLATINUM } Membership;

Membership membership = NONE;
```

- membership 변수의 등급 비교

```
membership == SILVER
```

※ 멤버십 등급을 출력하는 코드

➔ 코드 4-3
➔ 실행 결과

2. 조건 표현식

논리 연산자

- &&, ||, ! 연산자 기호 사용
- 각각 and, or, not 논리 연산을 나타냄

표 4-3 A && B와 A || B 연산

피연산자 A	피연산자 B	A && B	A || B
0(거짓)	0(거짓)	0(거짓)	0(거짓)
0(거짓)	0이 아닌 값(참)	0(거짓)	1(참)
0이 아닌 값(참)	0(거짓)	0(거짓)	1(참)
0이 아닌 값(참)	0이 아닌 값(참)	1(참)	1(참)

※ 멤버십별 무료 주차 가능성 평가 코드
(멤버십이 있다면 2시간 무료 주차 가능)

➔ 코드 4-4
➔ 실행 결과

2. 조건 표현식

단축 평가

- 논리 연산식을 평가할 때 왼쪽부터 순서대로 평가해 전체 연산식의 결괏값을 알 수 있으면 중단

표 4-4 단축 평가 조건

논리 연산식	단축 평가 조건	설명
A && B	A가 거짓	A가 거짓일 때 B 표현식은 평가(실행)되지 않는다.
A || B	A가 참	A가 참일 때 B 표현식은 평가(실행)되지 않는다.

```
(n != 0 && d / n > 0)
```

- n과 d가 변수라고 가정

```
(포인터_변수 != NULL && 포인터_변수의_메모리 주소를 이용해서 변수에 접근)
```

- 포인터 변수 사용 - 의사 코드 형태

03

조건문

3. 조건문

if 문

- 조건식을 평가해서 결괏값이 참(True)이 되면 정해진 코드를 실행
- 조건이 만족되지 않으면 더 이상 아무것도 하지 않음

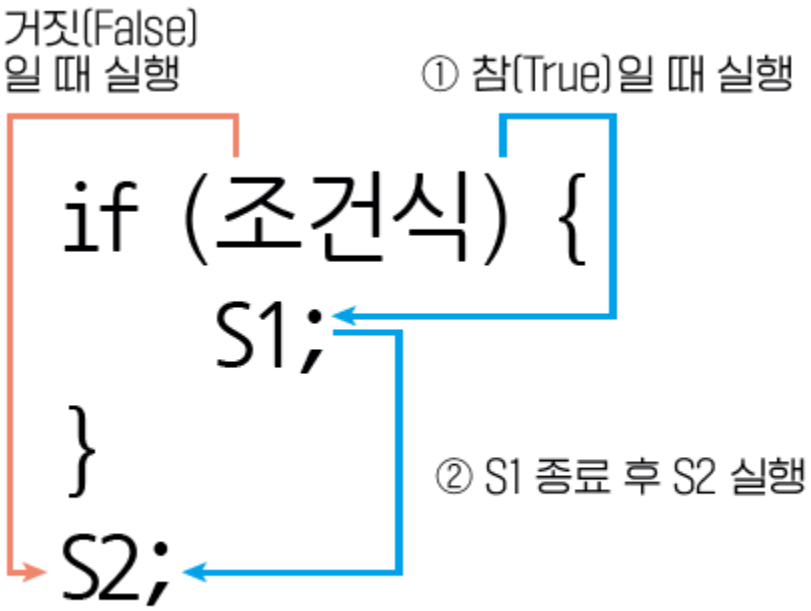

그림 4-1 if 문 동작 방법

3. 조건문

if 문

- 코드 블록 없이 작성한 if 문

코드 4-6

```
1    if (조건식)
2        S1;
3    S2;
```

- 다양한 유형별 if 문
  – 들여쓰기나 줄바꿈 등은 차이를 만들지 않음

유형 1

```
if (조건식)
{
 S1;
}
S2;
```

유형 2

```
if (조건식) { S1; }
S2;
```

유형 3

```
if (조건식) { S1; } S2;
```

3. 조건문

if 문

- 영화 예매 예시

표 4-5 좌석별 영화 관람료

좌석 열 이름	가격
A, B	13,000
C-K	14,000
L	15,000

코드 4-7 Movie1.c

```
1   #define _CRT_SECURE_NO_WARNINGS
2   #include <stdio.h>
3
4   int main(void)
5   {
6       char seat = '0';
7       printf("극장 좌석 열을 입력하세요: ");
8       scanf("%c", &seat);
9       if (seat == 'L') {
10          printf("표를 구매합니다\n");
11      }
12      return 0;
13  }
```

seat이라는 변수를 char 형으로 만들고 '0'으로 초기화

사용자로부터 문자를 한 개 입력받아서 seat에 저장
scanf() 대신 getchar() 함수 이용 가능. seat = (char) getchar();

seat가 'L'열이면 "표를 구매합니다\n" 문자열을 화면에 출력
그 외는 아무것도 하지 않음

3. 조건문

if 문

- 'L'을 입력했을 때

〈실행 결과〉

```
극장 좌석 열을 입력하세요: L
표를 구매합니다
```

- 'L'이 아닌 다른 열을 입력했을 때

〈실행 결과〉

```
극장 좌석 열을 입력하세요: H
```

※ 멤버십과 주차 시간을 입력받고, 조건이 만족되면 무료 주차가 가능하다고 출력하는 프로그램

➔ 코드 4-8
➔ 실행 결과

3. 조건문

if-else 문

- 조건이 만족될 때와 만족되지 않을 때 사용
- S1과 S2는 보통 한 줄 이상의 명령문을 나타냄

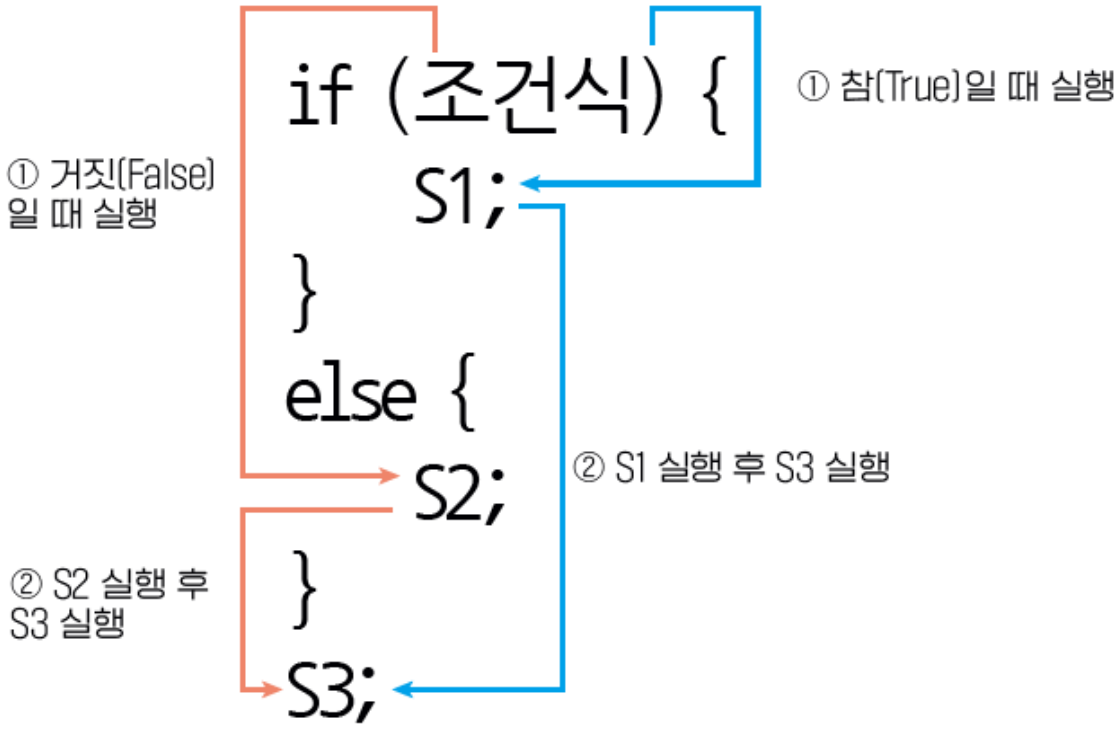

그림 4-2 if-else if-else 문 동작 방법

3. 조건문

if-else 문

- AI 스피커에게 오늘 비가 올 확률을 물어볼 때 답하는 코드 - 작성 조건

  - 강수 확률이 30%를 넘으면 우산을 가져갈 수 있도록 답하기
  - 아니면 가져가지 않아도 된다고 답하기
  - chanceOfRain은 강수 확률을 나타내는 변수
  - if-else 문을 작성할 때는 chanceOfRain 변수에 무작위로 0~100 사이의 정수를 생성해서 저장

※ AI 스피커에게 오늘 비가 올 확률을 물어볼 때 답하는 코드 - if 문

➔ 코드 4-9
➔ 실행 결과

※ AI 스피커에게 오늘 비가 올 확률을 물어볼 때 답하는 코드 - if-else 문

➔ 코드 4-10
➔ 실행 결과

3. 조건문

- **if 문이나 else 문에서 실행할 문장이 한 개만 있다면?**
  - 코드 블록 생략 가능

```
if (조건식)
    SingleStatement1;
else
    SingleStatement2;
```

```
if (조건식) {
    S1;
}
else
    SingleStatement2;
```

```
if (조건식)
    SingleStatement1;
else {
    S2;
}
```

3. 조건문

잘못 자리 잡은 else(dangling else) 문제

- 사용자가 1 이상의 정수를 입력했을 때 짝수이면 짝수라고 출력
- 1 미만의 값이 입력되면 잘못 입력했다고 알리고 종료

코드 4-11 EvenNumber.c

```
1   #define _CRT_SECURE_NO_WARNINGS
2   #include <stdio.h>
3
4   int main(void)
5   {
6       int n;
7       printf("짝수인지 확인할 1 이상의 정수 한 개를 입력하세요: ");
8       scanf("%d", &n);
9       if (n >= 1)
10          if (n % 2 == 0)
11              printf("n = %d는 짝수\n", n);
12      else
13          printf("1 미만의 정수가 입력되었습니다. 프로그램을 종료합니다\n");
14      return 0;
15  }
```

n이 1 이상인지 확인

n이 짝수이면 짝수라고 출력
아니면 아무것도 출력하지 않음

n이 1 미만이면, 잘못 입력했다고 출력

3. 조건문

■ 잘못 자리 잡은 else(dangling else) 문제

- 짝수 입력

〈실행 결과〉

```
짝수인지 확인할 1 이상의 정수 한 개를 입력하세요: 2
n = 2는 짝수
```

- 홀수 입력

〈실행 결과〉

```
짝수인지 확인할 1 이상의 정수 한 개를 입력하세요: 3
1 미만의 정수가 입력되었습니다. 프로그램을 종료합니다
```

3. 조건문

■ 잘못 자리 잡은 else(dangling else) 문제

```
if (n >= 1)
        if (n % 2 == 0)
            printf("n = %d는 짝수\n", n);
else
   printf("1 미만의 정수가 입력되었습니다. 프로그램을 종료합니다\n");
```

그림 4-3 else가 잘못 연결된 경우

3. 조건문

■ 잘못 자리 잡은 else(dangling else) 문제

- 해결 방법: 코드 블록을 사용

코드 4-12 EvenNumber2.c

```
1   #define _CRT_SECURE_NO_WARNINGS
2   #include <stdio.h>
3
4   int main(void)
5   {
6       int n;
7       printf("짝수인지 확인할 1 이상의 정수 한 개를 입력하세요: ");
8       scanf("%d", &n);
9       if (n >= 1) {
10          if (n % 2 == 0)
11              printf("n = %d는 짝수\n", n);
12      }
13      else
14          printf("1 미만의 정수가 입력되었습니다. 프로그램을 종료합니다\n");
15      return 0;
16  }
```

3. 조건문

■ 잘못 자리 잡은 else(dangling else) 문제

- 제대로 연결된 출력 결과

〈실행 결과〉

```
짝수인지 확인할 1 이상의 정수 한 개를 입력하세요: 3
```

3. 조건문

if-else if-else 문

- 세 가지 이상 중 한가지만 선택해서 실행할 때 사용

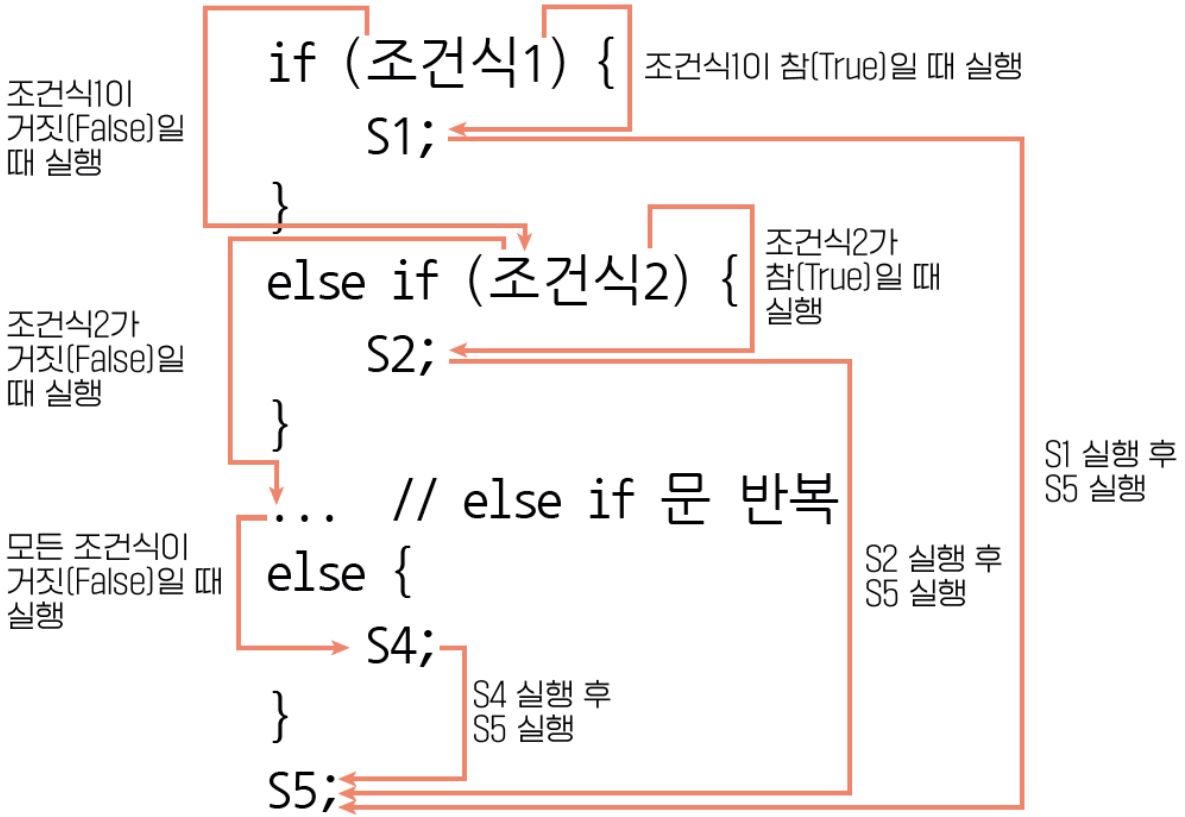

그림 4-4 if-else if-else 문 동작 방법

3. 조건문

if-else if-else 문

- else 없이 if-else if 문만 사용하는 것도 가능

```
if (조건식1) {
    S1;
}
else if (조건식2) {
    S2;
}
...  // 또 다른 조건식에 대한 else if 문 반복
S3;
```

3. 조건문

if-else if-else 문

- 극장에서 표를 구매하는 코드 완성하기
  - 사용자가 입력한 좌석의 열이 'A' 또는 'B'면 관람료는 13,000원
  - 'C'~'K' 사이면 14,000원
  - 'L'이면 15,000원
  - 그외의 글자가 입력된 경우 "좌석을 잘못 입력했습니다"
  - A열 또는 B열인 경우에 대해서 코드 작성
  - 사용자가 입력한 좌석의 열은 seat이라는 변수에 저장되어 있다고 가정

3. 조건문

if-else if-else 문

- 극장에서 표를 구매하는 코드 완성하기

```
if (seat == 'A') {
    printf("관람료는 13,000원입니다\n");
}
if (seat == 'B') {
    printf("관람료는 13,000원입니다\n");
}
```

- 논리 연산자 사용

코드 4-14

```
if (seat == 'A' || seat == 'B') {
    printf("관람료는 13,000원입니다\n");
}
```

코드 4-15 Movie2.c

```c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    char seat = '0';
    printf("극장 좌석 열을 입력하세요: ");
    scanf("%c", &seat);
    if (seat == 'A' || seat == 'B') {
        printf("%c열의 관람료는 13,000원입니다\n", seat);
    }
    else if (seat >= 'C' && seat <= 'K') {
        printf("%c열의 관람료는 14,000원입니다\n", seat);
    }
    else if (seat == 'L') {
        printf("%c열의 관람료는 15,000원입니다\n", seat);
    }
    else {
        printf("좌석을 잘못 입력했습니다\n");
    }
    return 0;
}
```

seat 변수 생성하고 초기화. 아무 값이나 저장해도 상관 없음

사용자로부터 좌석 열 입력받음. scanf()에 seat의 주소를 전달하는 것을 유의

seat 변숫값이 'A'나 'B'인지 확인
seat == 'A' || 'B'라고 작성하지 않게 주의

seat 변수가 'A', 'B'가 아니면 실행됨. 아스키 코드에서 'C'~'K'가 순차적으로 있으므로 크기 비교를 사용함

seat 변숫값이 'L'인지 확인

seat 변숫값이 'A'~'L'이 아니면 실행

<실행 결과>

```
극장 좌석 열을 입력하세요: A
A열의 관람료는 13,000원입니다

극장 좌석 열을 입력하세요: K
K열의 관람료는 14,000원입니다

극장 좌석 열을 입력하세요: L
L열의 관람료는 15,000원입니다
```

04

중첩 조건문

4. 중첩 조건문

- 조건문 안에 다른 조건문을 포함하는 것

코드 4-8 중첩 반복문으로 수정하기

- 기존 코드

```
if ((membership == 'G' || membership == 'P') && parkingTime <= 120) {
    printf("멤버십: %c, 주차시간: %d분. 무료 주차 가능\n", membership, parkingTime);
}
```

- 변경 코드

```
if (membership == 'G' || membership == 'P') { // GOLD or PLATINUM membership
    // 주차 시간 확인
    if (parkingTime <= 120) {
        printf("멤버십: %c, 주차시간: %d분. 무료 주차 가능\n", membership, parkingTime);
    }
}
```

4. 중첩 조건문

- 다른 경우의 코드 작성하기
  - 주차 요금을 계산할 때 멤버십이 SILVER(S) 또는 FRIENDS(F)이면 구매 금액을 입력받기
  - 30,000원 이상이면 2시간 무료, 10,000원 이상이면 1시간 무료 주차 가능

1. 멤버십이 'S' 또는 'F'여야 한다.
2. 1번이 만족되면 구매 금액을 입력받는다.
3. 금액이 3만원 이상이면 주차 시간이 120분 이하일 때 무료 주차가 가능하다.
4. 금액이 1만원 이상이면 주차 시간이 60분 이하일 때 무료 주차가 가능하다.

4. 중첩 조건문

- 다른 경우의 코드 작성하기

  - 1. 멤버십 확인하기

```
if (membership == 'S' || membership == 'F') { // 멤버십 확인
}
```

  - 2. 구매 금액 입력받기

```
int purchased = 0;
printf("구매 금액을 입력하세요: ");
scanf("%d", &purchased);
```

4. 중첩 조건문

- 다른 경우의 코드 작성하기

  - 3. 구매 금액이 3만원 이상이고, 주차시간이 120분 이하이면 무료 주차

```
if (purchased >= 30000 && parkingTime <= 120) {
    printf("멤버십: %c, 주차시간: %d분. 무료 주차 가능\n", membership, parkingTime);
}
```

  - 4. 구매 금액이 1만원 이상이고, 주차 시간이 60분 이하이면 무료 주차

```
if (purchased >= 10000 && parkingTime <= 60) {
    printf("멤버십: %c, 주차시간: %d분. 무료 주차 가능\n", membership, parkingTime);
}
```

※ 주차 요금 계산 중첩 조건문 코드 완성

➔ 코드 4-16

05

조건 연산자

5. 조건 연산자

- 조건 연산자 ?:는 세 개의 피연산자를 취하는 삼항 연산자
  - 연산자이므로 값을 생성
  - 결괏값은 조건식의 결과에 따라 표현식1 또는 표현식2의 결괏값

```
(조건식) ? 표현식1 : 표현식2
```

  - 변수에 대입하거나 함수에서 반환하는 형태로 사용

```
변수 = (조건식) ? 표현식1 : 표현식2;
return (조건식) ? 표현식1 : 표현식2;
```

  - 삼항 연산자 코드는 if-else 문으로 수정 가능

```
if (조건식) {
    변수 = 표현식1;
}
else {
    변수 = 표현식2;
}
```

5. 조건 연산자

- 극장 관람료 조조 관람 코드
  - 극장 관람료를 조조로 관람할 때 10,000원
  - 조조가 아닐 때 14,000원
  - matinee라는 변숫값이 0이면 조조 관람 아님/0이 아니면 조조 관람

```
int ticketPrice = (matinee) ? 10000 : 14000;
```

  - ticketPrice 변수에 matinee가 참이면 10000/거짓이면 14000 저장
  - 10000이나 14000처럼 정수 대신 표현식을 사용해도 됨

※ 기본 티켓 값 14,000원이고 입력한 문자가 'M'이면 조조 관람으로 4,000원 할인하는 코드

➔ 코드 4-17
➔ 실행 결과

06

switch 문

6. switch 문

- 여러 가지 중 한 가지를 선택하는 명령문
- switch 문으로 작성할 수 있는 코드는 if-else if-else 문

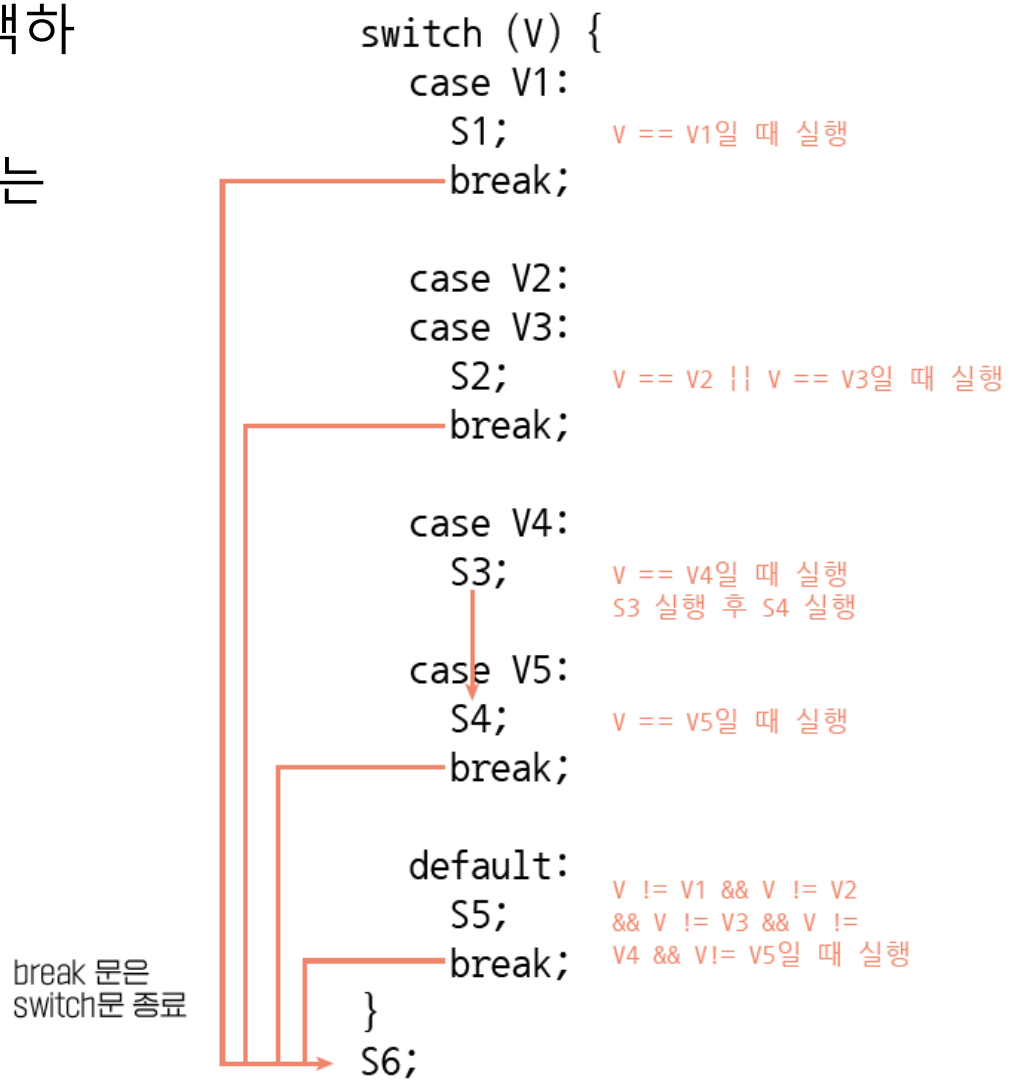

그림 4-5 switch 문 동작 방법

6. switch 문

- switch 문 사용 주의사항
  - switch 문의 변수는 정수형과 enum 자료형만 사용
  - case에 사용하는 V1, V2 등은 정수형 상수 또는 정수형 상수를 생성하는 표현식만 사용 가능
  - case 값은 중복이 불가능. 코드 4-18에서 V1~V5는 다른 값이어야 함
  - default는 생략 가능
  - case에 있는 S1이나 S2 등은 한 줄 이상의 명령문

6. switch 문

- switch 문을 if else if-else 문으로 변경

```
if (V == V1) {
    S1;
}
else if (V == V2 ¦¦ V == V3) {
    S2;
}
else if (V == V4) {
    S3;
    S4;
}
else if (V == V5) {
    S4;
}
else {
    S5;
}
S6;
```

※ 코드 4-15를 switch 문으로 변경하기

➔ 코드 4-19
➔ 실행 결과

6. switch 문

- 월(month)에 따라 31, 30, 28(29)일인지 화면에 출력하는 프로그램
  - 윤년이면 29일로 출력
  - 월은 enum 자료형으로 선언
  - JAN을 1로 지정해서 1월을 표현
  - 출력할 때 enum 값 그대로 사용 가능

```
typedef enum { JAN = 1, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC }
MONTH;
```

  - switch 문의 case 값으로 enum 값 지정 가능

```
MONTH month = JAN;
switch (month) {
    case JAN: case MAR:
        printf("%d월은 31일\n", month); // enum 값은 정수로 출력 가능
}
```

6. switch 문

- 월(month)에 따라 31, 30, 28(29)일인지 화면에 출력하는 프로그램
  - 윤년은 다음 조건으로 확인

> • 연도가 4로 나누어 떨어지면서 100으로 나누어 떨어지지 않는 해는 윤년(예 : 1984, 1544)
> • 연도가 400으로 나누어 떨어지는 해는 윤년

  - 조건을 코드로 작성하기

```
(year % 400 == 0) ¦¦ (year % 4 == 0 && year % 100 != 0)
```

※ 월에 따라 일을 출력하는 프로그램 완성

➔ 코드 4-20
➔ 실행 결과