



제06장

조건

단원 목표

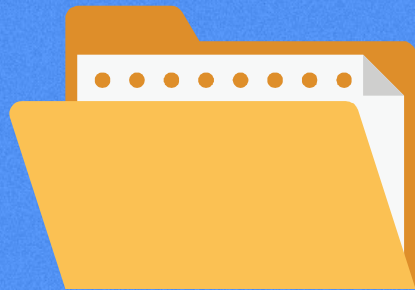
학습목표

- ▶ 제어문에 대하여 다음 용어를 이해하고 설명할 수 있다.
 - 순차적 실행과 제어문의 차이
 - 조건선택, 반복, 분기처리
- ▶ 조건에 따른 선택을 구현하는 if 문에 대하여 이해하고 구현할 수 있다.
 - if, if else, if else if, nested if 구문을 이해하고 샘플 소스 구현
 - 다양한 조건을 연산자로 구성
 - switch문을 이해하고 성적처리 구현
- ▶ 다양한 선택을 한 눈에 알아볼 수 있는 switch 문에 대하여 이해하고 구현할 수 있다.

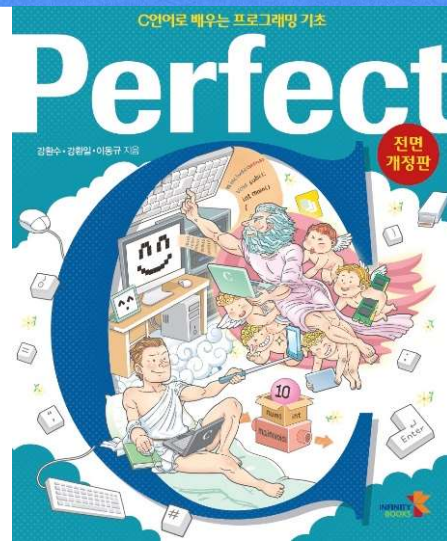
학습목차

- 6.1 제어문 개요
- 6.2 조건에 따른 선택 if 문
- 6.3 다양한 선택 switch 문





01. 제어문 개요



제어문의 종류

• 순차적 실행

- 지금까지 배워 온 프로그램의 실행 순서의 원칙
 - 순차적(sequential) 실행
- main 함수 내부에서 배치된 문장이 순차적으로 실행되는 흐름

• 비순차적 실행의 제어문

- 순차적 실행만으로 프로그램을 모두 작성한다면 매우 비효율적
- 프로그램의 실행 순서를 제어하는 제어문(control statement) 제공
 - 선택과 반복 등 순차적인 실행을 변형

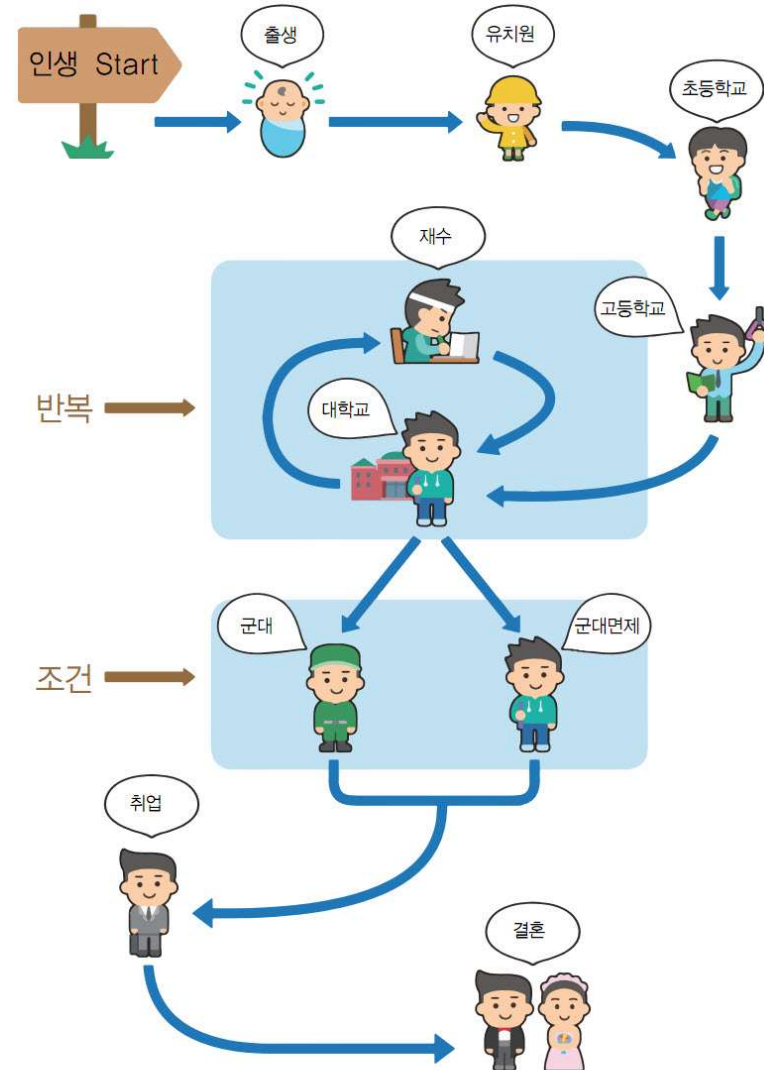


그림 6-1 인생여정과 프로그램의 실행 흐름

제어문 종류

• 조건선택

- 두 개 또는 여러 개 중에서 한 개를 선택하도록 지원하는 구문

• 반복(순환)

- 정해진 횟수 또는 조건을 만족하면 정해진 몇 개의 문장을 여러 번 실행하는 구문

• 분기처리

- break 문
 - 작업을 수행 도중 조건에 따라 반복이나 선택을 빠져 나가기
- continue 문
 - 일정구문을 실행하지 않고 다음 반복을 실행
- goto 문
 - 지정된 위치로 이동
- return 문
 - 작업 수행을 마치고 이전 위치로 돌아가는 구문

조건선택

조건에 대한 선택 구문

- if
- if else
- if else if
- nested if
- switch

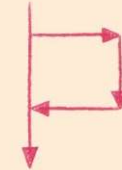


그림 6-2 조건선택

반복 순환

반복조건에 따라 일정영역의 반복 구문

- for
- while
- do while

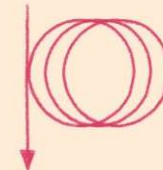


그림 6-3 반복순환

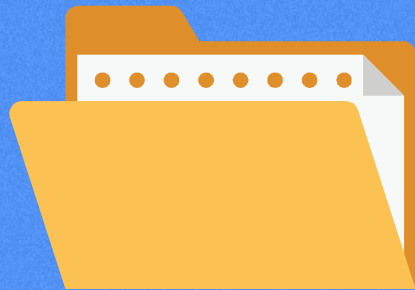
분기처리

지정된 영역으로 실행을 이동하는 구문

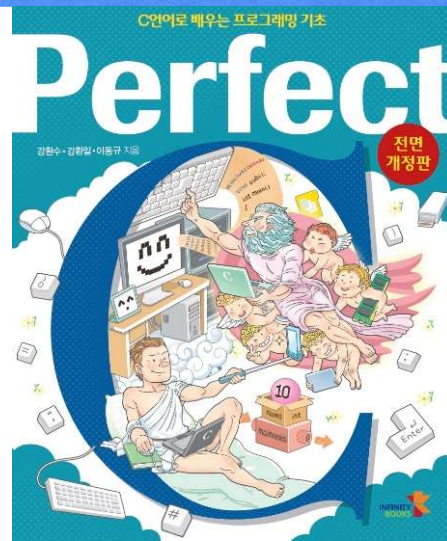
- break
- continue
- goto
- return



그림 6-4 분기처리



02. 조건에 따른 선택 if 문



조건에 따른 선택 개요

조건에 따른 결정

- 일상생활에서 조건에 따라 해야 할 내용이 결정되는 사례는 매우 많을 것
- 여러분은 성적에 따라: 조건
 - 장학금을 받을 수도 있고 못 받을 수도 있음

평균평점 ≥ 3.5

어느 학교는 평균평점이 3.5는 넘어야 장학금을 받을 수 있다.



석차 $\leq 0.05 * \text{학생수}$

또 우리 학교는 학과 석차가 상위 5%이어야 장학금을 받을 수 있다고 한다.

그림 6-5 조건에 따라 장학금이 결정

표 6-1 조건에 따라 선택이 발생하는 일상생활에서의 사례

조건 선택의 예	기준변수	조건 표현의 의사코드
온도가 32도 이상이면 폭염 주의를 출력	온도 temperature	만일 (temperature ≥ 32) printf("폭염 주의");
낮은 혈압이 100이상이면 '고혈압 초기'로 진단	혈압 low_pressure	만일 (low_pressure ≥ 100) printf("고혈압 초기");
속도가 40km와 60km 사이이면 "적정 속도"라고 출력	속도 speed	만일 (40 $\leq$ speed && speed $\leq$ 60) printf("적정 속도");
운전면허 필기시험에서 60점 이상이면 합격, 아니면 불합격 출력	시험 성적 point	만일 (point ≥ 60) printf("면허시험 합격"); 아니면 printf("면허시험 불합격");
남성이면 체력 테스트에서 80이상이면 합격이고, 아니면 불합격, 여성이면 70이상이면 합격, 아니면 불합격	여성, 남성 type 체력 점수 point	만일 남성이면 (type == 1) 만일 (point ≥ 80) printf("남성: 합격"); 아니면 printf("남성: 불합격"); 아니고 만일 여성이면 (type == 2) 만일 (point ≥ 70) printf("여성: 합격"); 아니면 printf("여성: 불합격");

if 문장(1)

- 조건에 따른 선택 if 문장

- 조건에 따른 선택을 지원하는 구문
- 일상의 예: 길의 선택
 - 만일 "비가 온다면" 개울이 없는 길 A 선택
 - "비가 오지 않는다면" 개울이 있지만 지름길인 길 B로 갈 것
- 바로 이러한 조건에 따른 문장이 if문

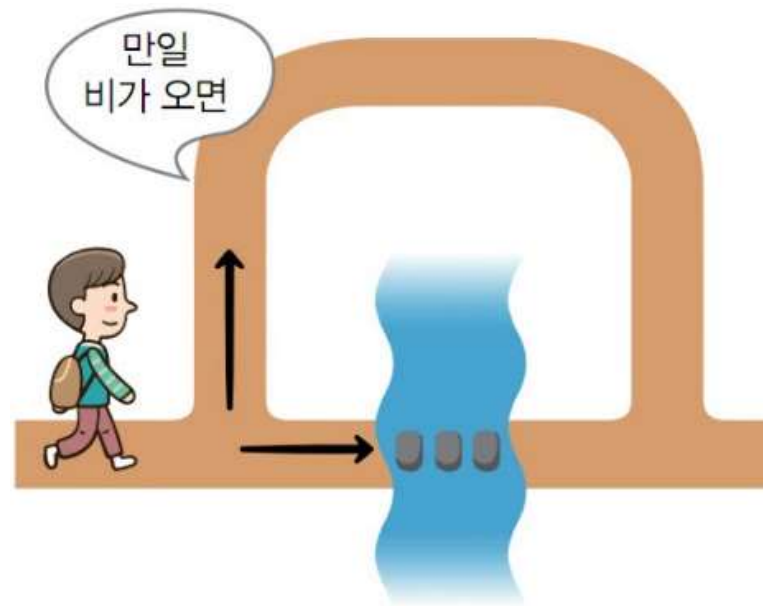


그림 6-6 길의 선택

if 문장(2)

가장 간단한 if문의 형태

- if (cond) stmt;
 - 조건식 cond가 0이 아니면(참)
 - stmt를 실행
 - 0이면(거짓)
 - stmt를 실행하지 않음
 - 문장 stmt는 여러 문장이라면 블록으로 구성
 - if 문이 종료되면
 - 그 다음 문장 실행
 - 조건식 (cond)는 반드시 괄호 필요
 - 문장 stmt은 반드시 들여쓰기

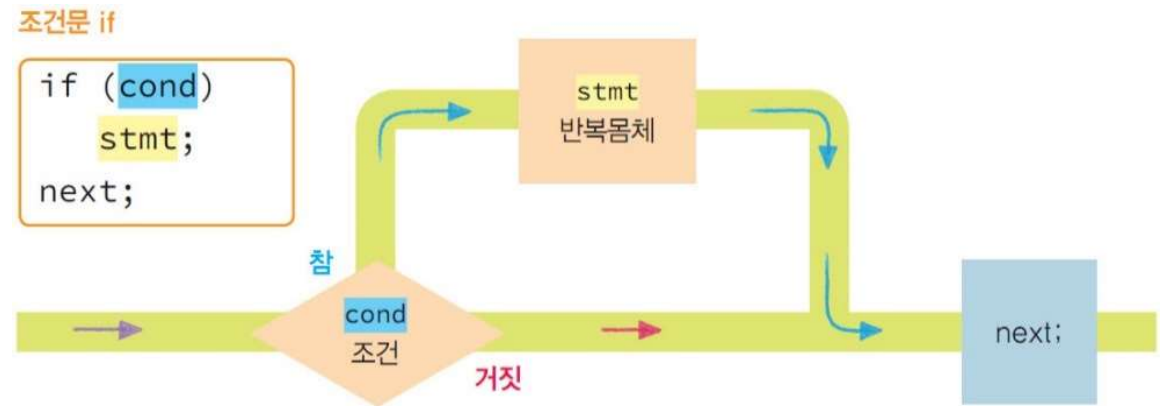


그림 6-7 if문과 두 개의 길

블록이면 다음과 같이 블록 시작 {을 if열에 맞추고 조건 문장들을 들여쓰기 마지막에 행에 다시 블록 종료 }를 시작 열에 맞춘다.

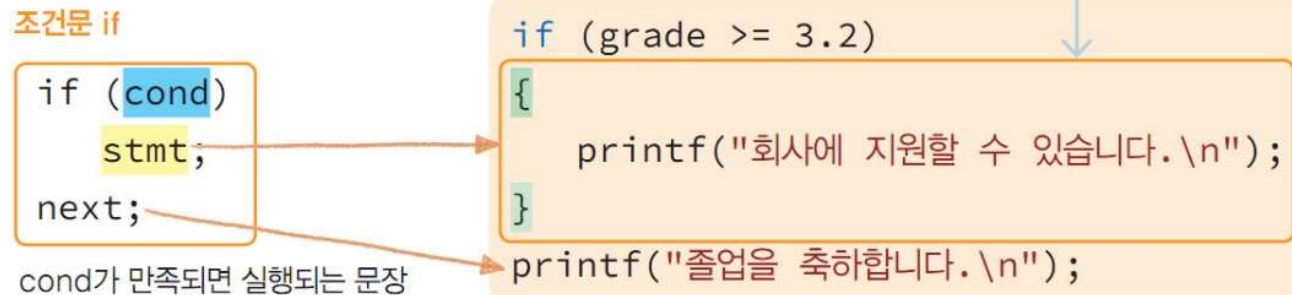


그림 6-8 if 문

조건 if문

예제 if.c

- 입력하는 온도에 따라 다른 출력

주의사항

- if (cond); stmt;와 같이 조건식 다음에 세미콜론을 쓰는 경우
 - 문법오류가 없으며 if 문의 조건으로 인해 실행되는 문장은 하나도 없고, 이후 stmt는 무조건 실행
 - 실제 코딩과정에서 문법오류가 발생하지 않아 논리적인 문제를 찾기 쉽지 않음
- if cond stmt;와 같이 조건식에 괄호를 빼먹는 경우
 - 문법오류가 표시되어 바로 수정할 수 있음

실습예제 6-1 if.c

```
01 // file: if.c
02 #define _CRT_SECURE_NO_WARNINGS
03
04 #include <stdio.h>
05
06 int main(void)
07 {
08     double temperature;
09
10     printf("현재 온도 입력: ");
11     scanf("%lf", &temperature);
12
13     if (temperature >= 32.0)
14     {
15         printf("폭염 주의보를 발령합니다.\n");
16         printf("건강에 유의하세요.\n");
17     }
18     printf("현재 온도는 섭씨 %.2f 입니다.\n", temperature);
19
20     return 0;
21 }
```

다음과 같이 블록 { 의 시작을 조건식 오른쪽으로 하기도 함

```
if (temperature >= 32.0) {
    printf("폭염 주의보를 발령합니다.\n");
    printf("건강에 유의하세요.\n");
}
```

설명

14~17 조건식이 만족한 경우, 여러 문장을 실행하기 위해서는 잊지 말고 중괄호의 블록을 사용
15~16 조건식 (temperature >= 32.0)의 의미인 입력된 온도가 32도 보다 높으면 두 문장이 실행
18 현재 온도를 출력하는 문장은 항상 실행

실행결과

현재 온도 입력: 31.3 현재 온도는 섭씨 31.30 입니다.	현재 온도 입력: 34.678 폭염 주의보를 발령합니다. 건강에 유의하세요. 현재 온도는 섭씨 34.68 입니다.
--	--

학점이 3.2 미만이라도 다음 두 문장은 항상 실행되는 결과를 낳는다.

```
if (grade >= 3.2);
    printf("회사에 지원할 수 있습니다.\n");

printf("졸업을 축하합니다.\n");
```

if 문에서 조건식 grade >= 3.2에는 괄호가 필요하다. 문법오류로 인한 오류통선이 발생하여 쉽게 알 수 있다.

```
if grade >= 3.2
    printf("회사에 지원할 수 있습니다.\n");

printf("졸업을 축하합니다.\n");
```

그림 6-9 if 문에서 세미콜론과 조건식 괄호에 주의

if else 문장(1)

- 조건 만족 여부에 대한 선택 if else
 - 조건문 if (cond) stmt1; else stmt2;
 - 조건 cond를 만족하면
 - stmt1을 실행
 - 조건 cond를 만족하지 않으면
 - stmt2를 실행
 - stmt1과 stmt2 둘 중의 하나를 선택하는 구문
- 조건식: 나머지 연산자 %의 사용
 - 조건식 (n % 2)
 - 0이 아니면(참) 홀수이고
 - 0이면(거짓) 짝수

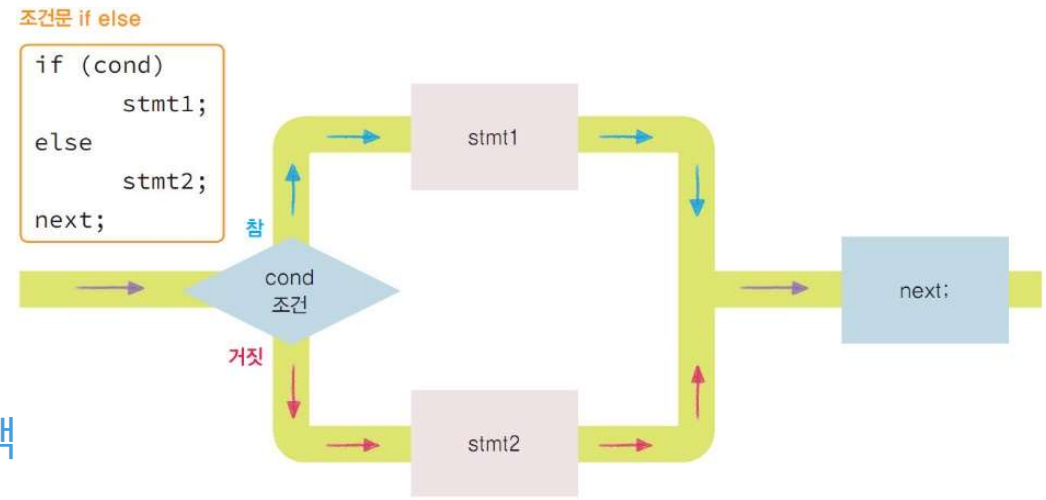


그림 6-10 if else문의 제어흐름

```
if (cond)
    stmt1;
else
    stmt2;
next;
```

```
if (n % 2 == 0)
    printf("짝수");
else
    printf("홀수");
printf("입니다.\n");
```

```
if (n % 2)
    printf("홀수");
else
    printf("짝수");
printf("입니다.\n");
```

그림 6-11 조건문 if else

if else 문장(2)

- 조건문 if else에서 주의해야 할 점
 - (조건식)은 괄호가 필요
 - 조건식에서 등호를 대입으로 잘못 쓰는 것에 주의
 - 즉 (n == 100)을 (n = 100)로 잘못 쓰면 항상 참으로 인식
 - stmt2 부분이 여러 문장이면 {여러 문장들}의 블록으로 구성
- 조건식에서 (n != 0)와 (n)은 같은 식
 - 마찬가지로 (n == 0)와 (!n)도 같은 식

표 6-2 if 문에 사용하는 조건식

조건식	설명	예	
(n != 0) (n)	n이 0이 아니어야 참인 연산식이므로 연산식 (n)과 같음	<pre>if (n % 2 != 0) printf("홀수"); else printf("짝수");</pre>	<pre>if (n % 2) printf("홀수"); else printf("짝수");</pre>
(n == 0) (!n)	n이 0이어야 참인 연산식이므로 연산식 (!n)과 같음	<pre>if (n % 2 == 0) printf("짝수"); else printf("홀수");</pre>	<pre>if (!(n % 2)) printf("짝수"); else printf("홀수");</pre>

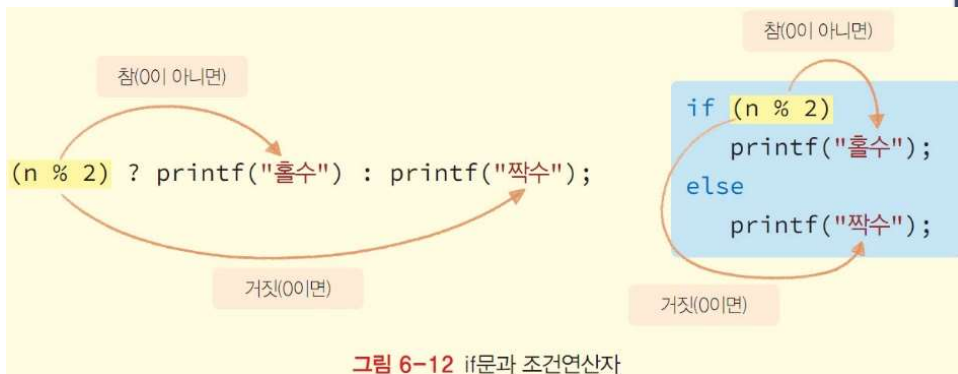
조건 if else 문

예제 ifelse.c

- 짝수와 홀수 구분

if와 조건연산자

- 조건 문장 if는 이미 배운 조건연산자와 매우 유사해서 좀 더 편리하게 활용



실습예제 6-2

ifelse.c

```
01 // file: ifelse.c
02 #define _CRT_SECURE_NO_WARNINGS
03
04 #include <stdio.h>
05
06 int main(void)
07 {
08     int n;
09
10     printf("정수 입력: ");
11     scanf("%d", &n);
12
13     if (n % 2) // if (n % 2 != 0)
14         printf("홀수");
15     else
16         printf("짝수");
17
18     printf("입니다.\n");
19
20     //조건연산자 이용
21     (n % 2) ? printf("홀수") : printf("짝수");
22     printf("입니다.\n");
23
24     return 0;
25 }
```

설명

14 조건식을 만족하면 실행할 문장이 하나이므로 블록으로 하지 않았으나 블록으로 구성해도 좋음
21 if문과 동일한 기능의 조건연산자

실행결과

정수 입력: 5
홀수입니다.
홀수입니다.

정수 입력: 6
짝수입니다.
짝수입니다.

반복된 조건에 따른 선택 if else if

조건문 if else 문

- else 이후에 if else를 필요한 횟수만큼 반복 가능
- 다음 문장에서
 - stmt1에서 stmt4에 이르는 여러 문장 중에서 실행되는 문장은 단 하나라는 것을 기억

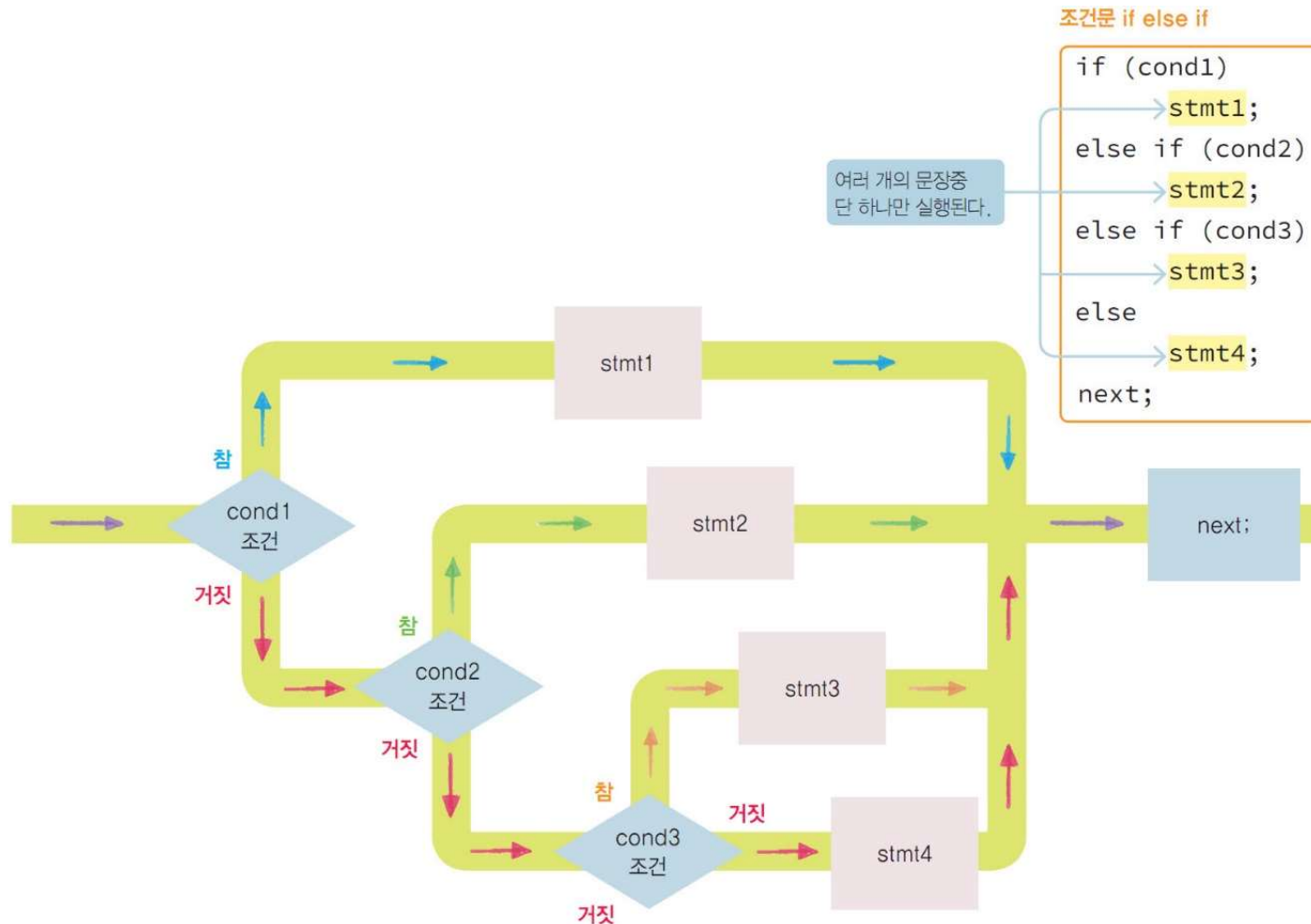


그림 6-13 조건문 if else if 문의 제어흐름

교과목 성적 처리

조건문 반복된 if else의 예

- 점수가 90이상이면 학점이 A
 - 80 이상에서 90 미만이면 B, 70 이상에서 80 미만이면 C
 - 60 이상에서 70 미만이면 D, 마지막으로 60 미만이면 F로 학점을 처리하는 모듈
- 가장 처음의 if는 if (point >= 90)으로 시작
- 그 다음 if는 else 이후의 if이므로 else if (point >= 80)으로 표현
 - 80 이상에서 90 미만인 조건(90 > point && point >= 80)이 만족

이 조건식은 첫 if의 조건식인 (point >= 90)이 만족되지 않고 체크되는 것이므로 결국 (!(point >= 90) && (point >= 80))이므로 80 이상에서 90 미만인 조건 (90 > point && point >= 80)이 만족된다.

```
if (point >= 90)
    printf("A\n");
else if (point >= 80)
    printf("B\n");
else if (point >= 70)
    printf("C\n");
else if (point >= 60)
    printf("D\n");
else
    printf("F\n");
```

필요하면 이와 같이 블록
사용이 가능하다.

```
if (point >= 90)
{
    printf("A\n");
}
else if (point >= 80)
{
    printf("B\n");
}
else if (point >= 70)
{
    printf("C\n");
}
else if (point >= 60)
{
    printf("D\n");
}
else
{
    printf("F\n");
}
```

조건 if else if 문

예제 ifelseif.c

- 평균평점의 점수에 따라 출력이 달라지는 예제

실습예제 6-4

ifelseif.c

```
01 // file: ifelseifelse.c
02 #define _CRT_SECURE_NO_WARNINGS
03
04 #include <stdio.h>
05
06 int main(void)
07 {
08     double gpa;
09
10     printf("평균평점 입력: ");
11
12     scanf("%lf", &gpa);
13
14     if (gpa >= 4.3)
15         printf("성적이 최고 우수한 학생입니다.\n");
16     else if (gpa >= 3.8)
17         printf("성적이 매우 우수한 학생입니다.\n");
18     else if (gpa >= 3.0)
19         printf("성적이 우수한 학생입니다.\n");
20     else
21         printf("성적이 3.0 미만인 학생입니다.\n");
22
23     return 0;
24 }
```

설명

11 자료형 double이면 형식지정자는 %lf
21 else if (gpa >= 3.8) 자체가 if (gpa < 4.3 && gpa >= 3.8)의 의미
20 이 문장은 위 조건식이 모두 거짓인 경우 실행되는 문장

실행결과

평균평점 입력: 4.3
성적이 최고 우수한 학생입니다.

평균평점 입력: 3.9
성적이 매우 우수한 학생입니다.

평균평점 입력: 3.3
성적이 우수한 학생입니다.

평균평점 입력: 2.7
성적이 3.0 미만인 학생입니다.

표 6-3 평균평점에 따른 메시지 출력

조건	단독 조건식	출력
평균평점 >= 4.3	<code>gpa >= 4.3</code>	성적이 최고 우수한 학생입니다.
4.3 > 평균평점 >= 3.8	<code>gpa < 4.3 && gpa >= 3.8</code>	성적이 매우 우수한 학생입니다.
4.3 > 평균평점 >= 3.0	<code>gpa < 3.8 && gpa >= 3.0</code>	성적이 우수한 학생입니다.
3.0 > 평균평점	<code>gpa < 3.0</code>	성적이 3.0 미만인 학생입니다.

LAB 표준입력으로 받은 두 실수의 대소에 따라 다양한 연산을 수행하여 그 결과를 출력

- 두 실수를 입력 받아 두 실수의 연산값이 출력되는 프로그램

- 만일 $x > y$ 이면 x / y 연산값 출력
- 만일 $x < y$ 이면 $x + y$ 연산값 출력
- 만일 $x == y$ 이면 $x * y$ 연산값 출력

- 결과

- 두 실수를 입력:
32.765 3.987
- 연산 결과: 8.22

Lab 6-1	tworeal.c
01	// file: tworeal.c
02	#define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의
03	
04	#include <stdio.h>
05	
06	int main(void)
07	{
08	double x = 0, y = 0, result = 0;
09	
10	printf("두 실수를 입력: ");
11	scanf("%lf %lf", &x, &y);
12	
13	-----
14	{
15	result = x / y;
16	}
17	else if (x == y)
18	{
19	result = x * y;
20	}
21	-----
22	{
23	result = x + y;
24	}
25	
26	printf("연산 결과: %.2f\n", result);
27	
28	return 0;
29	}
정답	13 if (x > y) 21 else

중첩된 if 문

예제 nestedif.c

- if 문 내부에 if문이 존재하면 중첩된 if문

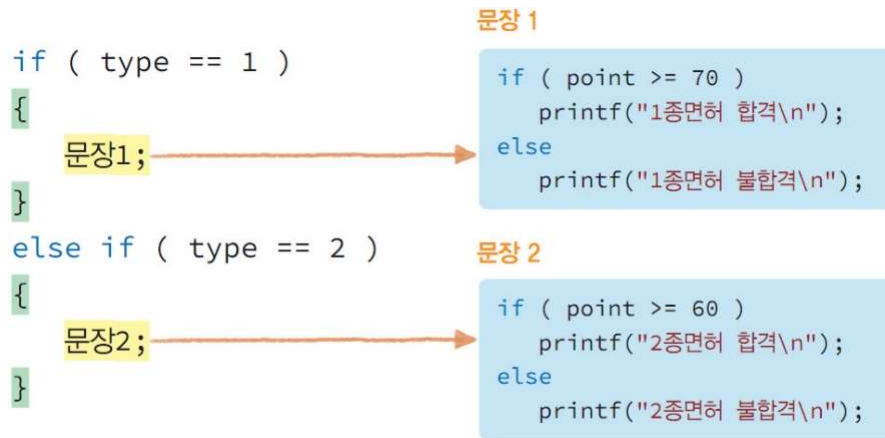


그림 6-15 중첩된 if

실습예제 6-5

nestedif.c

```
01 // file: nestedif.c
02 #define _CRT_SECURE_NO_WARNINGS
03
04 #include <stdio.h>
05
06 int main(void)
07 {
08     int type, point;
09
10     printf("번호를 선택: 1(1종면허), 2(2종면허): ");
11     scanf("%d", &type);
12     printf("필기시험 점수 입력: ");
13     scanf("%d", &point);
14
15     if (type == 1)
16     {
17         if (point >= 70)
18             printf("1종면허 합격\n");
19         else
20             printf("1종면허 불합격\n");
21     }
22     else if (type == 2)
23     {
24         if (point >= 60)
25             printf("2종면허 합격\n");
26         else
27             printf("2종면허 불합격\n");
28     }
29
30     return 0;
31 }
```

설명

11 변수 type에는 면허의 종류를 입력 저장
13 변수 point에는 면허 시험 성적을 입력 저장
15 외부 if 문
17~20 1종 면허인 경우, 점수를 합격여부를 판단하는 내부 조건문
24~27 2종 면허인 경우, 점수를 합격여부를 판단하는 내부 조건문

실행결과

번호를 선택: 1(1종면허), 2(2종면허): 1
필기시험 점수 입력: 67
1종면허 불합격

번호를 선택: 1(1종면허), 2(2종면허): 1
필기시험 점수 입력: 77
1종면허 합격

번호를 선택: 1(1종면허), 2(2종면허): 2
필기시험 점수 입력: 58
2종면허 불합격

번호를 선택: 1(1종면허), 2(2종면허): 2
필기시험 점수 입력: 63
2종면허 합격

블록 표시와 else

- 블록이 없는 경우 else 문장이 어느 if문의 소속인지 정확히 판단

- else는 문법적으로 같은 블록 내에서 else가 없는 가장 근접한 상위의 if문에 소속된 else로 해석

- else의 혼란을 방지하려면 블록을 이용하는 것을 권고

- 예제

- else는 두 번째 if (age >= 65)의 else
- else는 문법적으로 문제가 없으나
 - 나이가 20 이상 65 미만도 미성년으로 구분되니 논리적으로 잘못
 - 조건식 (age >= 20) 이후에 블록을 반드시 사용한 후 else를 배치

```
if ( type == 1 )
{
    if ( point >= 70 )
        printf("1종면허 합격\n");
    else
        printf("1종면허 불합격\n");
}
else if ( type == 2 )
{
    if ( point >= 60 )
        printf("2종면허 합격\n");
    else
        printf("2종면허 불합격\n");
}
```

```
if ( type == 1 )
{
    if ( point >= 70 )
        printf("1종면허 합격\n");
    else
        printf("1종면허 불합격\n");
}
else if ( type == 2 )
{
    if ( point >= 60 )
        printf("2종면허 합격\n");
    else
        printf("2종면허 불합격\n");
}
```

그림 6-16 if에서의 적절한 블록의 사용

```
int age = 30;
if (age >= 20)
    if (age >= 65)
        printf("당신은 어르신입니다.\n");
    else
        printf("당신은 미성년자입니다.\n");
```

```
int age = 30;
if (age >= 20)
{
    if (age >= 65)
        printf("당신은 어르신입니다.\n");
    else
        printf("당신은 미성년자입니다.\n");
}
```

```
int age = 30;
if (age >= 20)
{
    if (age >= 65)
        printf("당신은 어르신입니다.\n");
}
else
    printf("당신은 미성년자입니다.\n");
```

그림 6-17 else 구문의 주의

다양한 if문

일상생활에서의 사례와 구현

표 6-4 조건에 따라 선택이 발생하는 일상생활에서의 사례와 구현

조건 표현	if 형태	기준변수	다양한 if 문으로 구성
온도가 32도 이상이면 폭염주의를 출력	if	온도 temperature	<code>if (temperature >= 32)</code> <code>printf("폭염주의");</code>
속도가 40km와 60km 사이이면 "적정속도"라고 출력	if	속도 speed	<code>if (40 <= speed && speed <= 60)</code> <code>printf("적정속도");</code>
운전면허 필기시험에서 60점 이상이면 합격, 아니면 불합격 출력	if else	시험성적 point	<code>if (point >= 60)</code> <code>printf("면허시험 합격");</code> <code>else</code> <code>printf("면허시험 불합격");</code>

속도 speed가 40km와 60km 사이라는 조건식

– `(40 <= speed && speed <= 60)`

- `(40 <= speed <= 60)`로 사용하면 잘못된 조건식

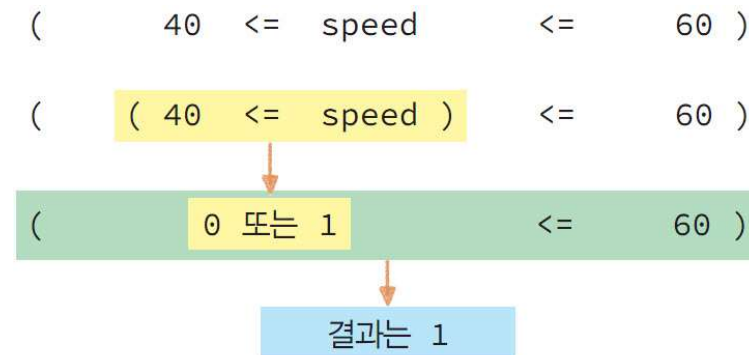


그림 6-18 주의해야 할 조건 연산식

if와 조건연산자

- 조건연산자의 기능은 if문으로도 가능
 - 간단히 조건연산자로 구현할 수 있는 기능은 조건연산자를 사용하는 것이 더 효율적

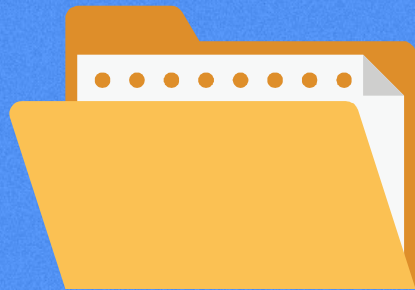
표 6-5 조건연산자와 if

구현 내용	조건연산자	if
두 수의 최대값 구하기	<code>max = x > y ? x : y;</code>	<pre>if (x > y) max = x; else max = y;</pre>
두 수의 최소값 구하기	<code>min = x > y ? y : x;</code>	<pre>if (x > y) min = y; else min = x;</pre>
절대값 구하기	<code>abs = x >= 0 ? x : -x;</code>	<pre>if (x >= 0) abs = x; else abs = -x;</pre>
홀수와 짝수 구하기	<code>a % 2 ? printf("홀수") : printf("짝수");</code>	<pre>if (a % 2) printf("홀수"); else printf("짝수");</pre>

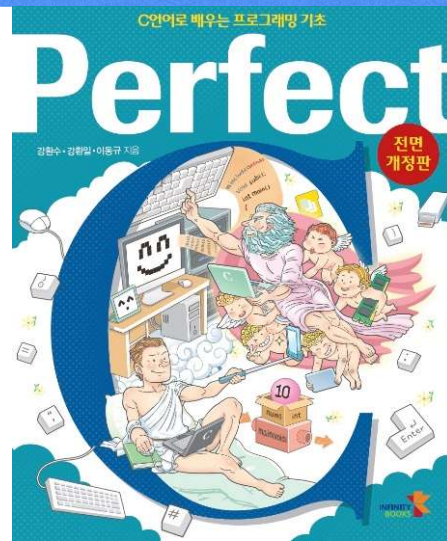
LAB 표준입력으로 받은 세 정수의 최대값을 출력

- 표준입력으로 받은 세 정수에서 최대값이 출력되는 프로그램
 - 먼저 조건식 $x > y$ 이 참이면 x 와 y 의 최대값, 거짓이면 x 와 y 의 최소값
- 결과
 - 세 정수를 입력: 10 30 20
 - 최대 수: 30

Lab 6-2	maxof3.c
01	// file: maxof3.c
02	#define _CRT_SECURE_NO_WARNINGS
03	
04	#include <stdio.h>
05	
06	int main(void)
07	{
08	int x, y, z;
09	
10	printf("세 정수를 입력: ");
11	scanf("%d %d %d", &x, &y, &z);
12	
13	if (x > y)
14	{
15	if (_____)
16	printf("최대수: %d\n", x);
17	else
18	printf("최대수: %d\n", z);
19	}
20	else
21	{
22	if (_____)
23	printf("최대수: %d\n", y);
24	else
25	printf("최대수: %d\n", z);
26	}
27	
28	return 0;
29	}
정답	15 if (x > z) 22 if (y > z)



03. 다양한 선택 switch 문



switch 문장 개요

• 연산식의 정수 또는 문자 선택

- 다중선택 구문인 switch문을 사용
 - 문장 if else가 여러 번 계속 반복되는 구문을 좀 더 간략하게 구현
 - 특히 if의 조건식이 정수와 등호식이라면 보다 간편한 switch문의 사용이 가능
- 주어진 연산식이 문자형 또는 정수형
 - 그 값에 따라 case의 상수값과 일치하는 부분의 문장들을 수행하는 선택 구문

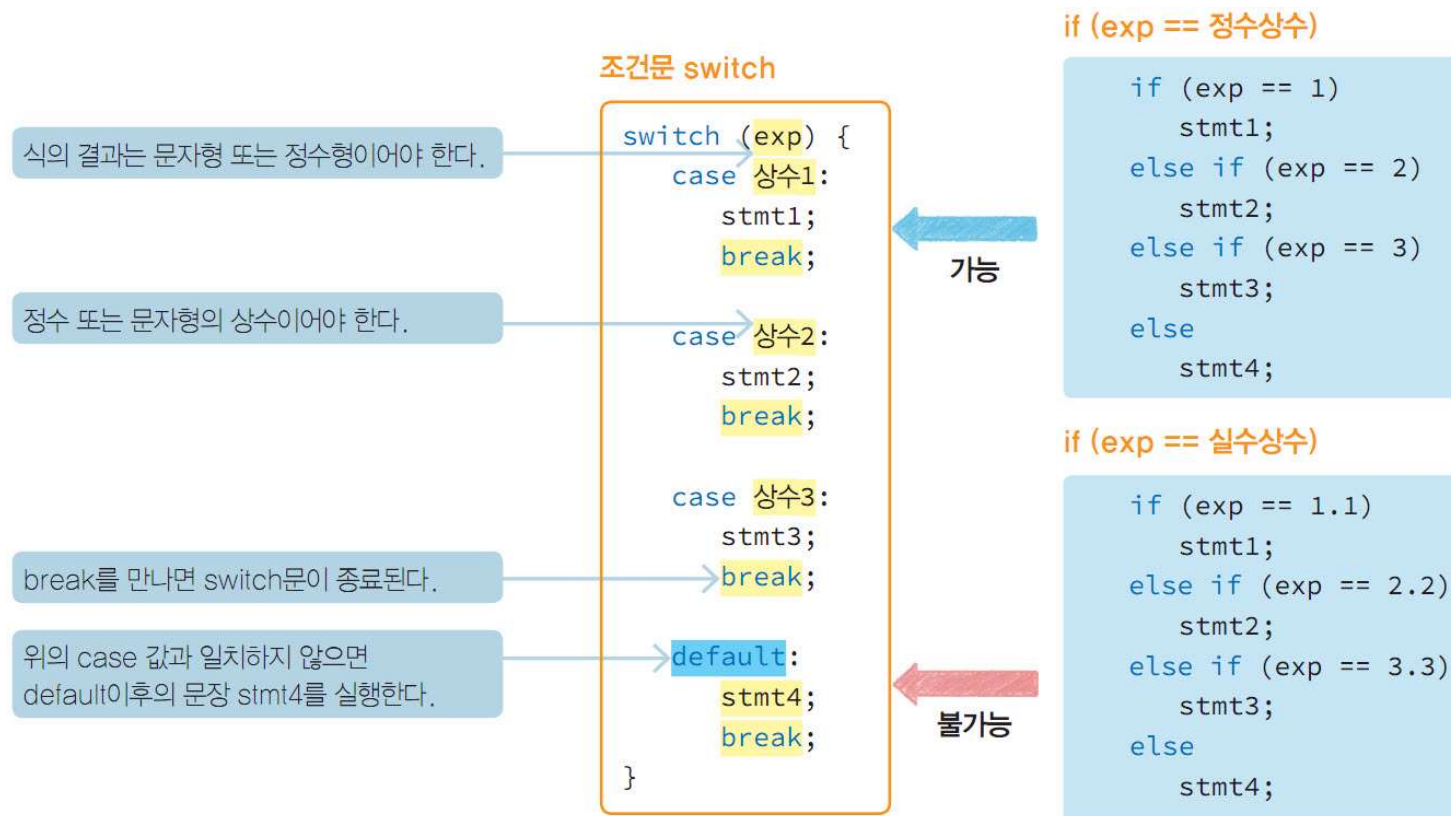


그림 6-19 switch 문 문장구조

switch 문

- **switch (exp) { ...} 문: switch, case, break, default는 키워드**
 - 표현식 exp 결과값: 반드시 문자 또는 정수
 - case의 값과 일치하는 항목의 문장 stmt1을 실행한 후 break를 만나 종료
 - case 다음의 value 값은 변수가 올 수 없으며
 - 상수식(constant expression)으로 그 결과가 정수 또는 문자 상수
 - 값은 중복 불가능
 - break 문이 없으면
 - break 문을 만나기 전까지 다음 case 의 내부로 무조건 이동하여 내부 문장을 실행

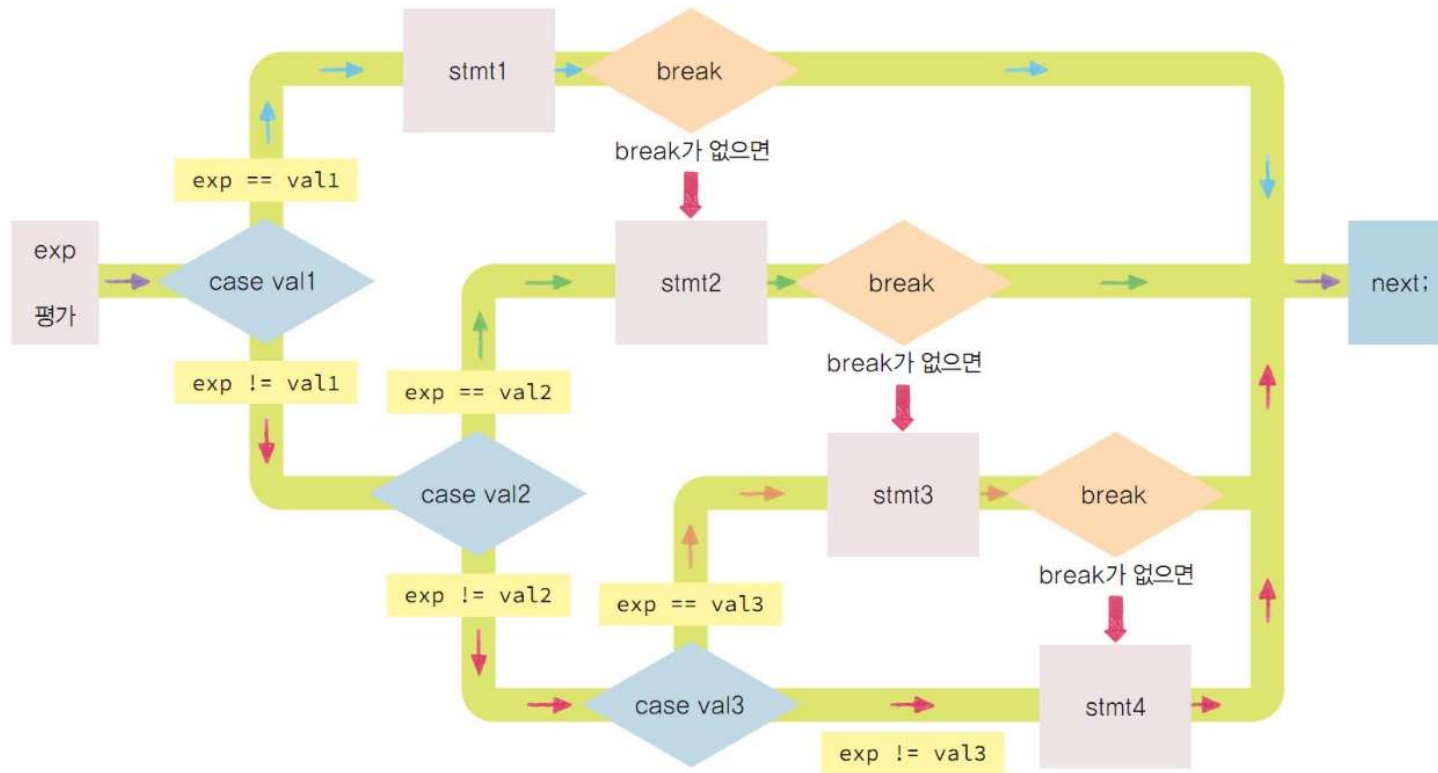


그림 6-20 switch 문의 제어흐름

switch 문

예제 switch.c

- 1에서 4까지 정수 중에서 선택한 번호에 따라 표준 입력한 두 실수의 더하기, 빼기, 곱하기 나누기를 실행하는 프로그램

28 break를 생략하면 논리오프로 바로 다음 default인 “번호를 잘못 선택했습니다.”를 출력
32 입력한 연산종류가 1에서 4번이 아니라도 그 이후에 실행할 문장이 없으므로 break 생략 가능

실행결과

두 실수 입력: 3.765 6.987
연산종류 번호선택 1(+), 2(-), 3(*), 4(/): 1
3.77 + 6.99 = 10.75

두 실수 입력: 3.986 4.826
연산종류 번호선택 1(+), 2(-), 3(*), 4(/): 3
3.99 * 4.83 = 19.24

두 실수 입력: 4.82 3.987
연산종류 번호선택 1(+), 2(-), 3(*), 4(/): 2
4.82 - 3.99 = 0.83

두 실수 입력: 87.354 6.98
연산종류 번호선택 1(+), 2(-), 3(*), 4(/): 4
87.35 / 6.98 = 12.51

실습예제 6-7

switch.c

```
01 // file: switch.c
02 #define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의
03
04 #include <stdio.h>
05
06 int main(void)
07 {
08     double x, y, result;
09     int op;
10
11     printf("두 실수 입력: ");
12     scanf("%lf %lf", &x, &y);
13     printf("연산종류 번호선택 1(+), 2(-), 3(*), 4(/): ");
14     scanf("%d", &op);
15
16     switch (op) {
17     case 1:
18         printf("%.2f + %.2f = %.2f\n", x, y, x + y);
19         break;
20     case 2:
21         printf("%.2f - %.2f = %.2f\n", x, y, x - y);
22         break;
23     case 3:
24         printf("%.2f * %.2f = %.2f\n", x, y, x * y);
25         break;
26     case 4:
27         printf("%.2f / %.2f = %.2f\n", x, y, x / y);
28         break;
29
30     default:
31         printf("번호를 잘못 선택했습니다.\n");
32         break; //생략가능
33     }
34
35     return 0;
36 }
```

설명

12 scanf("두 실수 입력: %lf %lf", &x, &y); 이와 같이 쓰지 않도록 주의, 입력을 위한
프롬프트는 printf() 함수를 이용
19 break를 생략하면 논리오프로 바로 다음 case 2인 빼기도 수행
22 break를 생략하면 논리오프로 바로 다음 case 3인 더하기도 수행
25 break를 생략하면 논리오프로 바로 다음 case 4인 나누기도 수행

break의 적절한 사용

예제 seasonswitch.c

- 표준입력으로 1년 중 해당하는 월을 입력 받아 그 달에 맞는 계절을 출력하는 프로그램

OR 개념 처리

- case 문 내부에 break 문이 없다면
 - 일치하는 case 문을 실행하고
 - break 문을 만나기 전까지 다음 case 내부 문장을 실행
- case 4, 5와 같은 나열은 문법오류가 발생

```
switch ( month )
{
    case 4 : case 5 :
        printf("%d월은 봄입니다.\n", month);
        break;

    case 6 : case 7 : case 8 :
        printf("%d월은 여름입니다.\n", month);
        break;

    ...

    default :
        printf("월(month)을 잘못 입력하셨습니다.\n");
}
```

```
case 4, 5 : //오류발생
...
break;

case 6, 7, 8 : //오류발생
...
break;
```

그림 6-21 여러 개의 정수 선택에 따른 case 처리 방법

실습예제 6-8

seasonswitch.c

```
01 // file: seasonswitch.c
02 #define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의
03
04 #include <stdio.h>
05
06 int main(void)
07 {
08     int month;
09
10     printf("년도의 월(month)을 입력: ");
11     scanf("%d", &month);
12
13     switch (month) {
14         case 4: case 5:
15             printf("%d월은 봄입니다.\n", month);
16             break;
17
18         case 6: case 7: case 8:
19             printf("%d월은 여름입니다.\n", month);
20             break;
21
22         case 9: case 10: case 11:
23             printf("%d월은 가을입니다.\n", month);
24             break;
25
26         case 12: case 1: case 2: case 3:
27             printf("%d월은 겨울입니다.\n", month);
28             break;
29
30         default:
31             printf("월(month)을 잘못 입력하셨습니다.\n");
32     }
33
34     return 0;
35 }
```

설명

```
12 scanf("%d", &month);에 의해 변수 month에 월이 입력
14 입력값이 4월, 5월이면
15 "~ 봄입니다." 출력
16 switch 문 종료
18 입력값이 6월, 7월, 8월이면
19 "~ 여름입니다." 출력
20 switch 문 종료
22 9월, 10월, 11월이면
23 "~ 가을입니다." 출력
24 switch 문 종료
26 12월, 1월, 2월, 3월이면
27 "~ 봄입니다." 출력
```

LAB 표준입력으로 받은 세 정수의 최대값을 출력

- 표준입력으로 받은 세 정수에서 최대값이 출력되는 프로그램
 - 먼저 조건식 $x > y$ 의 결과를 switch 문을 이용
 - 조건연산자를 이용하여 두 수 중에서 최대값을 출력
- 결과
 - 세 정수를 입력: 5 10 8
 - 최대값: 10

Lab 6-3	simplemaxof3.c
	<pre>01 // file: simplemaxof3.c 02 #define _CRT_SECURE_NO_WARNINGS 03 04 #include <stdio.h> 05 06 int main(void) 07 { 08 int x, y, z; 09 10 printf("세 정수를 입력: "); 11 scanf("%d %d %d", &x, &y, &z); 12 13 switch ((x > y)) 14 { 15 case 0: 16 printf("최대값: %d\n", _____); 17 break; 18 19 case 1: 20 printf("최대값: %d\n", _____); 21 break; 22 } 23 24 return 0; 25 }</pre>
정답	<pre>16 printf("최대값: %d\n", y > z ? y : z); 20 printf("최대값: %d\n", x > z ? x : z);</pre>

switch 연산식의 활용

예제 scoreswitch.c

- 표준 입력된 성적에 따라 성적 'A'에서 'F'까지 부여
- 점수를 10으로 나눈 연산식 (score / 10)을 활용
- switch 문으로 성적처리 가능

실습예제 6-9

scoreswitch.c

```
01 // file: scoreswitch.c
02 #define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의
03
04 #include <stdio.h>
05
06 int main(void)
07 {
08     int score;
09
10     printf("점수 입력: ");
11     scanf("%d", &score);
12
13     switch (score / 10) {
14     case 10: case 9:
15         printf("점수가 %d 점으로 성적이 %c 입니다.\n", score, 'A');
16         break;
17     case 8:
18         printf("점수가 %d 점으로 성적이 %c 입니다.\n", score, 'B');
19         break;
20     case 7:
21         printf("점수가 %d 점으로 성적이 %c 입니다.\n", score, 'C');
22         break;
23
24     case 6:
25         printf("점수가 %d 점으로 성적이 %c 입니다.\n", score, 'D');
26         break;
27
28     default:
29         printf("점수가 %d 점으로 성적이 %c 입니다.\n", score, 'F');
30     }
31
32     return 0;
33 }
```

설명

13 연산식 (score / 10)은 정수의 나누기 결과는 정수라는 것을 활용
14 100점도 있으므로 case 10이 필요
16, 19, 22, 25 break를 생략하면 다음 case 내부 문장을 수행하므로 반드시 필요
27 연산값이 0, 1, 2, 3, 4, 5인 경우를 처리

실행결과

점수 입력: 100
점수가 100 점으로 성적이 A 입니다.

점수 입력: 88
점수가 88 점으로 성적이 B 입니다.

점수 입력: 75
점수가 75 점으로 성적이 C 입니다.

점수 입력: 62
점수가 62 점으로 성적이 D 입니다.

표 6-6 점수에 따른 성적처리를 위한 연산값

점수 예	점수 범위	(score / 10) 연산값	성적처리
100, 98, 95, 90	90 <= 점수 <= 100	9 또는 10	'A' 부여
80, 85, 88, 89	80 <= 점수 < 90	8	'B' 부여
80, 85, 88, 89	70 <= 점수 < 80	7	'C' 부여
80, 85, 88, 89	60 <= 점수 < 70	6	'D' 부여
30, 55, 58, 59	점수 < 60	그 외	'F' 부여

default의 위치

- 일반적으로 switch 문에서 default는 생략 가능

- default는 선택적
 - 일치된 case 값을 만나지 못하여 default를 만나면 default 내부의 문장을 실행
- default의 위치
 - 어디에도 위치 가능
 - 중간에 위치하면서 break문이 없으면 하부 case 내부 문장을 무조건 실행하므로 주의가 필요
- default를 위치시킨 이후에 다른 case가 있다면
 - break를 반드시 입력
- 만일 break문이 제거되면
 - case 상수 외에도 다시 상수 10과 9에 기술된 내부 문장이 실행되는 논리 오류 발생

```
switch (score / 10) {  
    default:  
        printf("점수가 %d 점으로 성적이 %c 입니다.\n", score, 'F');  
        break; //반드시 필요한 break  
  
    case 10: case 9:  
        printf("점수가 %d 점으로 성적이 %c 입니다.\n", score, 'A');  
        break;  
    ...  
}
```

그림 6-23 default의 위치에 따른 break의 추가

- 표준입력으로 받은 정수에 대응하는 열거 상수로 switch문에서 분기를 처리하는 프로그램

- 삼원색을 표현하는 열거상수로 RED, GREEN, BLUE를 정의
- 세 정수(R[0], G[1], B[2]) 중의 하나를 입력
- switch의 case 상수로 열거 상수를 이용

- 결과

- 세 정수(R[0], G[1], B[2]) 중의 하나를 입력: 0
- Red

실습예제 6-3	simplemaxof3.c
	<pre>01 // enumswitch.c 02 #define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의 03 04 #include <stdio.h> 05 06 int main(void) 07 { 08 enum color { RED, GREEN, BLUE }; 09 int input; 10 11 printf("세 정수(R[0], G[1], B[2]) 중의 하나를 입력: "); 12 scanf("%d", &input); 13 14 switch (input) { 15 case ____: 16 printf("Red\n"); 17 break; 18 19 case ____: 20 printf("Green\n"); 21 break; 22 23 case ____: 24 printf("Blue\n"); 25 26 break; 27 28 default: 29 printf("잘못된 입력\n"); 30 } 31 32 return 0; 33 }</pre>
정답	<pre>15 case RED: 19 case GREEN: 23 case BLUE:</pre>



Thank you