C프로그래밍

5 조건문 (2/2), for문

Review [예제 03]

- **문자를 입력받을 때 이전에 scanf( )를 호출했으면 공백을 추가**

```
08    printf("첫 번째 수를 입력하세요 : ");
09    scanf("%d", &a);                          ----- 계산할 첫 번째 숫자를 입력한다.
10    printf("계산할 연산자를 입력하세요 : ");
11    scanf(" %c", &ch);                        ----- 연산자를 입력한다.
12    printf("두 번째 수를 입력하세요 : ");
13    scanf("%d", &b);                          ----- 계산할 두 번째 숫자를 입력한다.
14
```

Review [예제 04]

- **If문을 중첩해서 사용 시 if~ else if~ else 문을 사용**

```
14
15    if(ch == '+')                                  ----- 중복 if문을 사용한
16      printf("%d + %d = %d 입니다. \n", a, b, a+b);      연산을 수행한다.
17    else if(ch == '-')
18      printf("%d - %d = %d 입니다. \n", a, b, a-b);
19    else if(ch == '*')
20      printf("%d * %d = %d 입니다. \n", a, b, a*b);
21    else if(ch == '/')
22      printf("%d / %d = %f 입니다. \n", a, b, a/(float)b);
23    else if(ch == '%')
24      printf("%d %% %d = %d 입니다. \n", a, b, a%b);
25    else                                           ----- +, -, *, /, % 외의
26      printf("연산자를 잘못 입력했습니다. \n");            문자를 입력하면
27  }                                                      오류 메시지를 보여준다.
```

if ~ else{ if ~ else }

➔ if ~ else if ~ else로 사용 가능

01
switch ~ case문

switch~case문 (1/5)

- **다중 분기 : 여러 개 중에 하나를 선택하는 것**
  - ✓ 다중 분기를 할 수 있는 방법 : if else if…. vs. switch ~ case
- **switch ~ case**
  - ✓ switch의 조건값(정숫값)에 따라 case문을 실행
  - ✓ 정숫값이 1이면 1의 값에 해당하는 문장, 정숫값이 2이면 2의 값에 해당하는 문장을 실행하는 방식

```
if(정숫값 == 1)
            실행할 문장1;

else if(정숫값 == 2)
            실행할 문장2;

else        실행할 문장3;
```

```
switch(정숫값){
  case 정숫값 1:
     실행할 문장 1;
     break;
  case 정숫값 2:
     실행할 문장 2;
     break;
  default:
     실행할 문장 3;
     break;
}
```

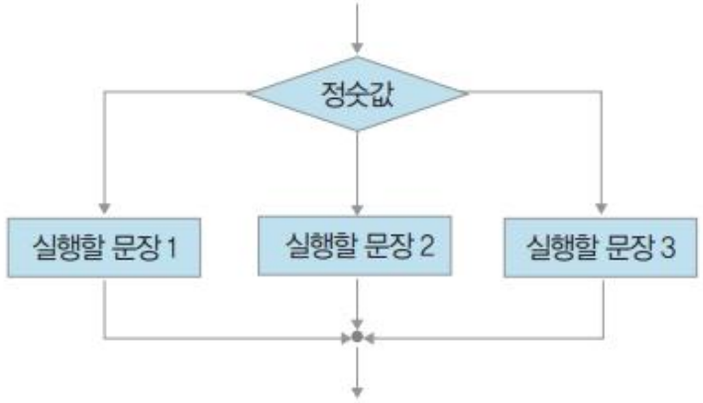


그림 5-7 switch~case문의 형식과 순서도

switch~case문 (2/5)

기본 5-9 switch~case문 사용 예 1 — 5-9.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int a;
06
07    printf("1 ~ 4 중에 선택하세요 : ");
08    scanf("%d", &a);
09
10    switch(a)                    ---- 입력한 a 값에 따라서 분기한다.
11    {
12    case 1:                      ---- a가 1이면 13행을 수행하고 14행에서 switch 블록을
13       printf("1을 선택했다\n");       빠져나간다.
14       break;
15    case 2:
16       printf("2를 선택했다\n");
17       break;
18    case 3:
19       printf("3을 선택했다\n");
20       break;
21    case 4:
22       printf("4를 선택했다\n");
23       break;
24    default:                     ---- a가 1, 2, 3, 4 중 어디에도 해당되지 않을 때
25       printf("이상한 걸 선택했다.\n");    수행된다.
26    }
27 }
```

실행 결과

```
1 ~ 4 중에 선택하세요 : 3
3을 선택했다
```

switch~case문 (3/5)

- **기본 5-9 복기**
  - ✓ 8행에서 입력한 숫자가 10행의 switch문으로 전달
  - ✓ 만약 10이면 12행으로, 20이면 15행으로, 30이면 18행으로, 40이면 21행으로…
  - ✓ 입력한 숫자가 1, 2, 3, 4가 아니면 24행의 default 부분 실행

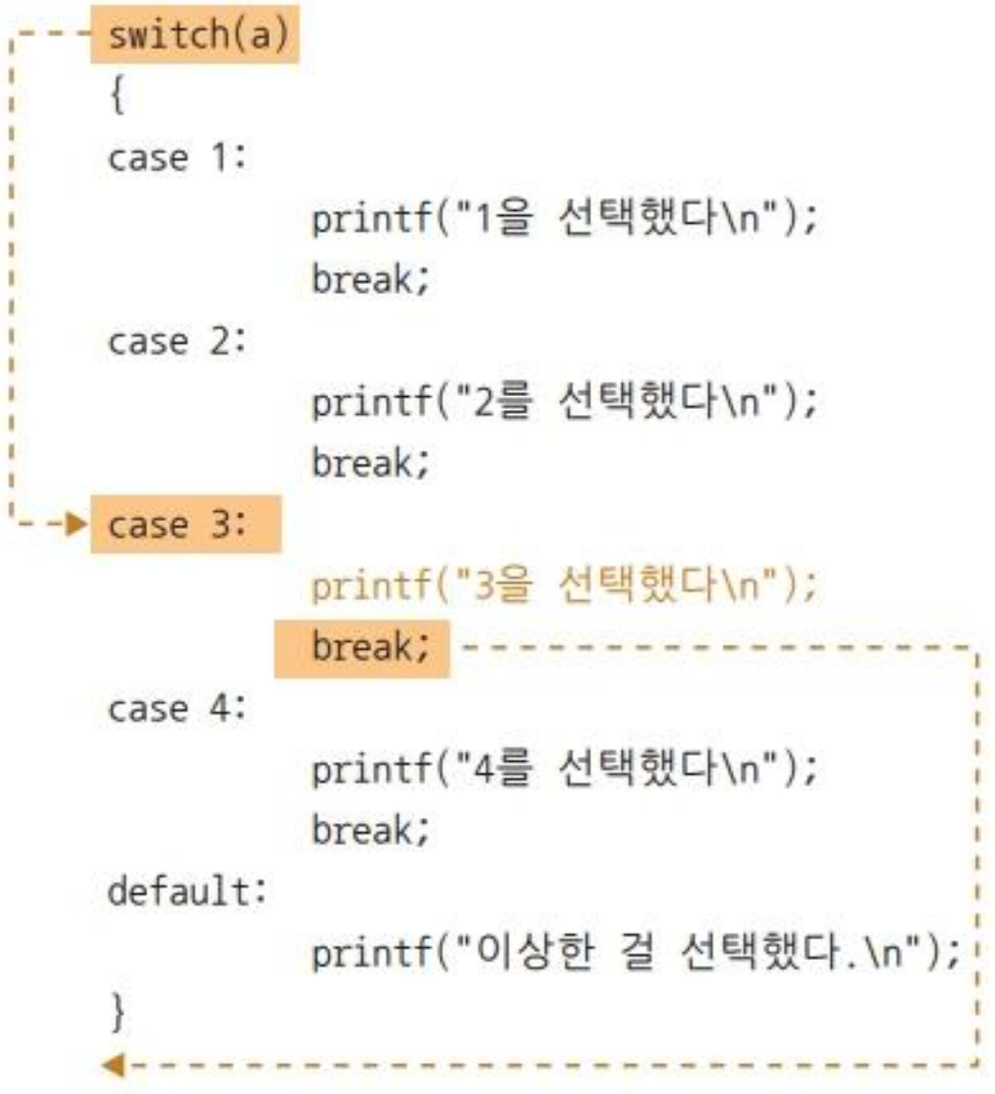


그림 5-8 a가 3일 때의 switch~case문 흐름도

switch~case문 (4/5)

- **기본 5-9 복기**
  - ✓ 각각의 실행문 마지막에 반드시 break문을 써야 한다는 것이 중요
  - ✓ 그렇지 않으면 switch~case문을 빠져나가지 못한 채 다음 코드를 계속 수행

```
10  switch(a)
11  {
12  case 1:
13    printf("1을 선택했다\n");
14  case 2:
15    printf("2를 선택했다\n");
16  case 3:
17    printf("3을 선택했다\n");
18  case 4:
19    printf("4를 선택했다\n");
20  default:
21    printf("이상한 걸 선택했다.\n");
22  }
```

실행 결과

```
1 ~ 4 중에 선택하세요 : 2
2를 선택했다
3을 선택했다
4를 선택했다
이상한 걸 선택했다.
```

switch~case문 (5/5)

응용 5-10 switch~case문 사용 예 2 — 5-10.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int  year;
06
07    printf("출생연도를 입력하세요 : ");
08    scanf("%d", &year);
09
10    switch( __1__ )
11    {
12    case 0 : printf("원숭이띠\n");   break;
13    case 1 : printf("닭띠\n");       break;
14    case 2 : printf("개띠\n");       break;
15    case 3 : printf("돼지띠\n");     break;
16    case 4 : printf("쥐띠\n");       break;
17    case 5 : printf("소띠\n");       break;
18    case 6 : printf("호랑이띠\n");   break;
19    case 7 : printf("토끼띠\n");     break;
20    case 8 : printf("용띠\n");       break;
21    case 9 : printf("뱀띠\n");       break;
22    case 10 : printf("말띠\n");      break;
23    case 11 : printf("양띠\n");      break;
24    }
25 }
```

정답_ 1 year % 12

입력한 연도를 12로 나눈 나머지값에 따라서 결과를 출력한다.

실행 결과

```
출생연도를 입력하세요 : 2003
양띠
```

02
단순 for문

for문의 이해 (1/2)

- **For문은 여러 번 수행해야 할 작업을 한 번에 해결해주는 반복문**
  - ✓ 반복문이란 말 그대로 실행 문장을 '반복'하게 만들어주는 것
  - ✓ 주로 for문과 while문이 사용

기본 6-1 같은 문장을 반복 출력하는 예 6-1.c

반복문을 사용하지 않는 경우

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    printf("안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^\n");   ----- 내용을 출력한다.
06    printf("안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^\n");
07    printf("안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^\n");
08    printf("안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^\n");
09    printf("안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^\n");
10 }
```

실행 결과

```
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
```

for문의 이해 (2/2)

기본 6-2 기본 for문 사용 예 6-2.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i;
06
07    for(i=0; i < 5 ; i++)
08    {
09       printf("안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^\n");
10    }
11 }
```

반복문을 사용하는 경우

for문을 사용해서 다섯 번 반복한다.

실행 결과

```
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
```

for문의 개념과 활용 (1/15)

- **for문의 형식**
  - ✓ 괄호 안에 초깃값, 조건식, 증감식이 세미콜론(;)으로 구분됨
  - ✓ 그리고 중괄호({ }) 안에 반복할 문장들이 있음
- **반복되는 순서 : ❶, ❷를 수행한 뒤 ❸, ❹, ❷ 순서로 반복**

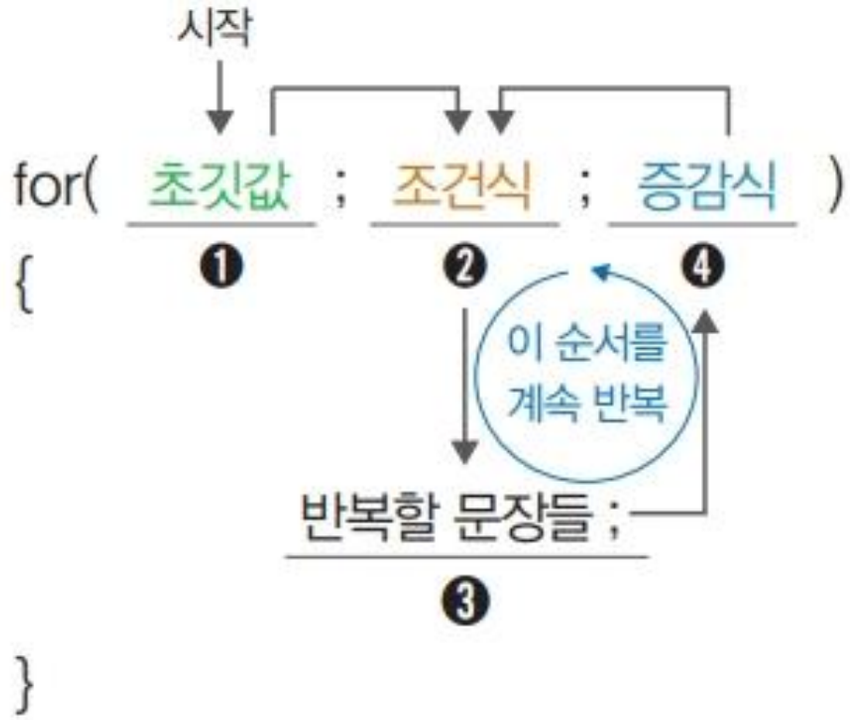


그림 6-1 for문의 기본 형식

for문의 개념과 활용 (2/15)

- **기본 6-2 복기**

그림 6-2 for문의 개념과 실제 사용

❶ for문을 사용하려면 무조건 변수를 하나 준비

❷ 사용할 변수의 초깃값을 0으로 설정

❸ i가 5가 될 때 까지 검사하기 위해 조건을 i < 5로 설정

❹ i++는 i=i+1과 동일한 역할

❺ 실제로 반복되는 내용

for문의 개념과 활용 (3/15)

- **for문이 반복되는 순서**

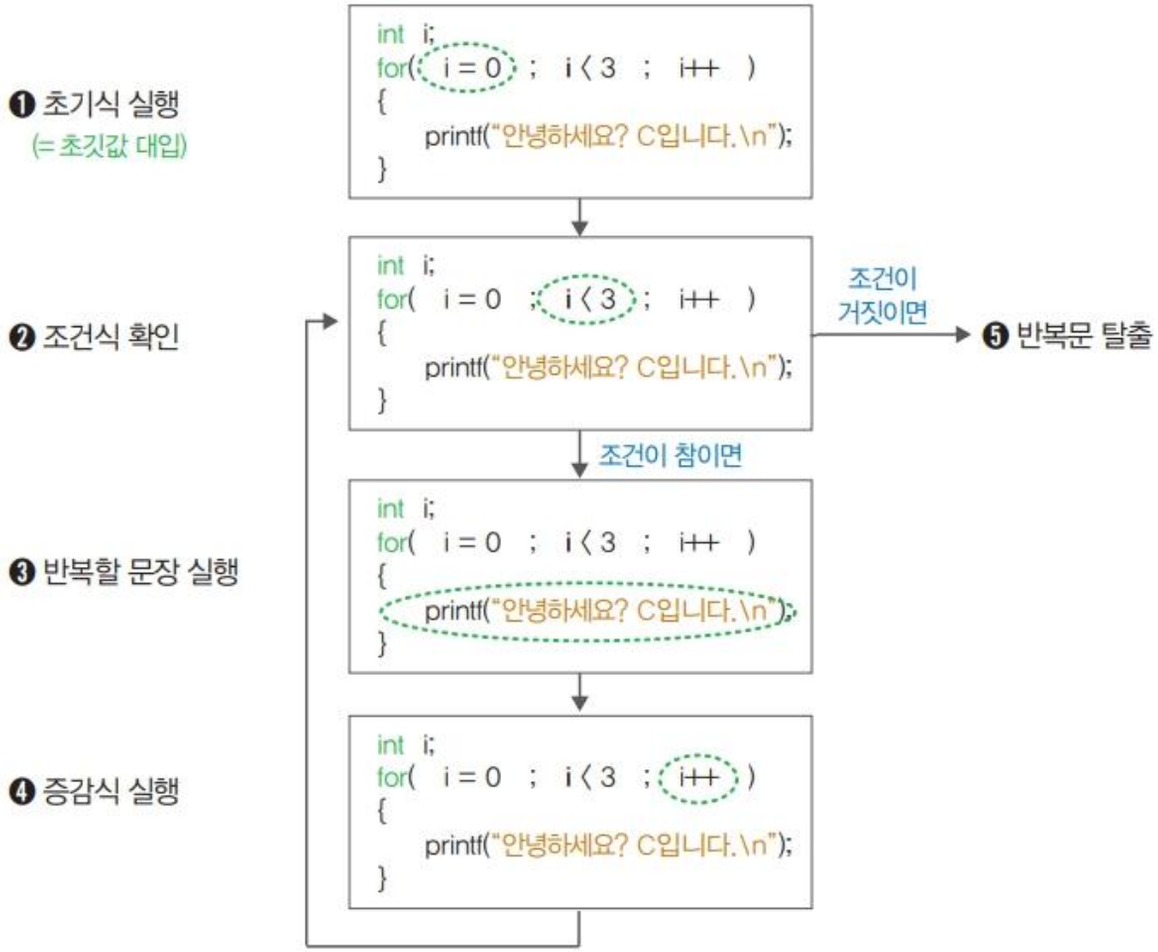


그림 6-3 for문이 반복되는 순서

for문의 개념과 활용 (4/15)

- **for문이 반복되는 순서 (cont'd)**

▶제1회: ❶ 초기식을 수행한다(현재 i=0).

▶제2회: ❷ 조건식을 확인한다. 현재 i 값이 0이므로 i < 3은 참이다.
▶제3회: ❸ printf문을 수행한다('안녕하세요? …'를 출력한다).
▶제4회: ❹ 증감식 i++를 수행해서 i 값을 1 증가시킨다(현재 i=1).

▶제5회: 다시 ❷ 조건식을 확인한다. 현재 i 값이 1이므로 i < 3은 참이다.
▶제6회: 다시 ❸ printf문을 수행한다('안녕하세요? …'를 출력한다).
▶제7회: 다시 ❹ 증감식 i++를 수행해서 i 값을 1 증가시킨다(현재 i=2).

▶제8회: 다시 ❷ 조건식을 확인한다. 현재 i 값이 2이므로 i < 3은 참이다.
▶제9회: 다시 ❸ printf문을 수행한다('안녕하세요? …'를 출력한다).
▶제10회: 다시 ❹ 증감식 i++를 수행해서 i 값을 1 증가시킨다(현재 i=3).

▶제11회: 다시 ❷ 조건식을 확인한다. 현재 i 값이 3이므로 드디어 i < 3이 거짓이다.
▶제12회: 조건식이 거짓이므로 ❺ 반복문을 탈출해 반복문 블록({ }) 밖의 내용을 수행한다.

for문의 개념과 활용 (5/15)

- **for문이 반복되는 순서 (cont'd)**
  - ✓ 초기식이 i = 0이 아니라 i = 3이라면
  - ➔ 조건식이 i < 3이므로 거짓이 되어 바로 ❺를 수행, printf문은 한 번도 수행되지 않음
  - ✓ 조건식을 확인한 후 증감식(i++)을 바로 수행하지 않고 **반복할 문장을 먼저 수행한 후 증감식을 수행**
  - ✓ 반복할 문장이 하나뿐이라면 중괄호를 생략해도 무관

```
int i;
for( i=0; i < 3; i++ )
   printf("빙글빙글 for문입니다. ^^\n");
```

==

```
int i;
for( i=0; i < 3; i++ )
{
   printf("빙글빙글 for문입니다. ^^\n");
}
```

for문의 개념과 활용 (6/15)

- **{ }의 사용 예**

기본 6-3 for문과 블록 사용 예 6-3.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i;
06    for( i=0; i < 3; i++ )
07    {
08       printf("안녕하세요? \n");
09       printf("##또 안녕하세요?## \n");
10    }
11
12    printf("\n\n");
13
14    for( i=0; i < 3; i++ )
15       printf("안녕하세요? \n");
16       printf("##또 안녕하세요?## \n");
17 }
```

----- 블록을 사용한 for문이다.

----- 블록을 사용하지 않은 for문이다.

실행 결과

```
안녕하세요?
##또 안녕하세요?##
안녕하세요?
##또 안녕하세요?##
안녕하세요?
##또 안녕하세요?##

안녕하세요?
안녕하세요?
안녕하세요?
##또 안녕하세요?##
```

for문의 개념과 활용 (7/15)

- **초깃값과 증감식을 다르게 설정하는 경우**

기본 6-4 for문 사용 예 1 6-4.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i;
06
07    for( i=5; i > 0; i-- )          ----- 초깃값, 조건식, 증감식을 수정한다.
08    {
09       printf("%d : 안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^\n", i);
10    }
11 }
```

실행 결과

```
5 : 안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
4 : 안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
3 : 안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
2 : 안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
1 : 안녕하세요? 빙글빙글 for문을 공부 중입니다. ^^
```

for문의 개념과 활용 (8/15)

- **for문의 변수를 활용하는 예**

기본 6-5 for문 사용 예 2 — 6-5.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i;
06
07    for( i=1; i <= 5; i++ )          ---- i 값이 1부터 5까지 변경된다.
08    {
09       printf("%d \n", i);
10    }
11 }
```

실행 결과

```
1
2
3
4
5
```

for문의 개념과 활용 (9/15)

- **for문의 활용 : 1부터 10까지의 합을 구하기**

기본 6-6 for문을 활용하지 않고 합계 구하기 6-6.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int hap;
06
07    hap = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10;      ----- hap에 1부터 10까지 더해서 입력한다.
08
09    printf(" 1에서 10까지의 합: %d \n", hap);
10 }
```

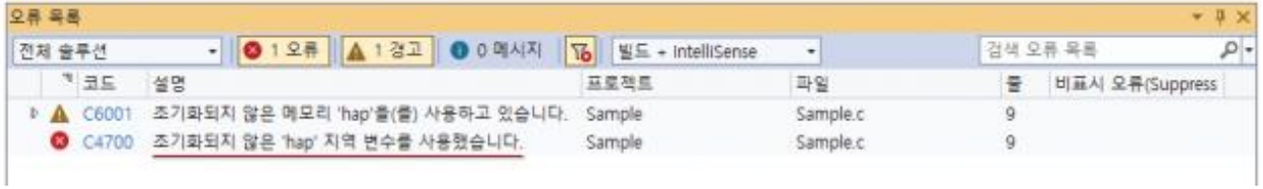
for문의 개념과 활용 (10/15)

- **for문의 활용 : 1부터 10까지의 합을 구하기 (cont'd)**

기본 6-7 for문을 활용하여 합계 구하기 1 6-7.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int hap;                    ----- 합계를 누적할 변수를 선언한다.
06    int i;                      ----- 1부터 10까지 변하는 변수를 선언한다.
07
08    for( i=1; i <= 10; i++ ) {  ----- for문에 의해 1부터 10까지 열 번 반복된다.
09       hap = hap + i;           ------- hap 변수에 1부터 10까지 반복해서 누적된다.
10    }
11
12    printf(" 1에서 10까지의 합: %d \n", hap);
13 }
```

실행 결과 ▼



for문의 개념과 활용 (11/15)

- **for문의 활용 : 1부터 10까지의 합을 구하기 (cont'd)**

기본 6-8 for문을 활용하여 합계 구하기 2 6-8.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int hap = 0;                  ---- 합계를 누적할 변수를 선언하고 0으로 초기화한다.
06    int i;
07
08    for( i=1; i <= 10; i++ ) {
09       hap += i;                  ----- hap 변수에 1부터 10까지 반복해서 누적한다.
10    }                                    hap = hap+i와 동일하다.
11
12    printf(" 1에서 10까지의 합: %d \n", hap);
13 }
```

for문의 개념과 활용 (12/15)

- **기본 6-8 복기**
  - ✓ 주의할 것은 변수 i와 hap의 값
  - ✓ 5행에서 hap을 0으로 초기화, 8행에서 i 값을 1로 초기화
  - ✓ 첫 번 째 반복에서 i 값 1을 현재 hap의 값인 0과 더해 다시 hap에 넣음
  - ✓ 두 번째 반복에서는 i 값 2를 현재 hap의 값인 1과 더해 다시 hap에 넣음
  - ✓ 이런 과정이 열 번 반복되면 1~10을 더한 값이 hap에 들어감

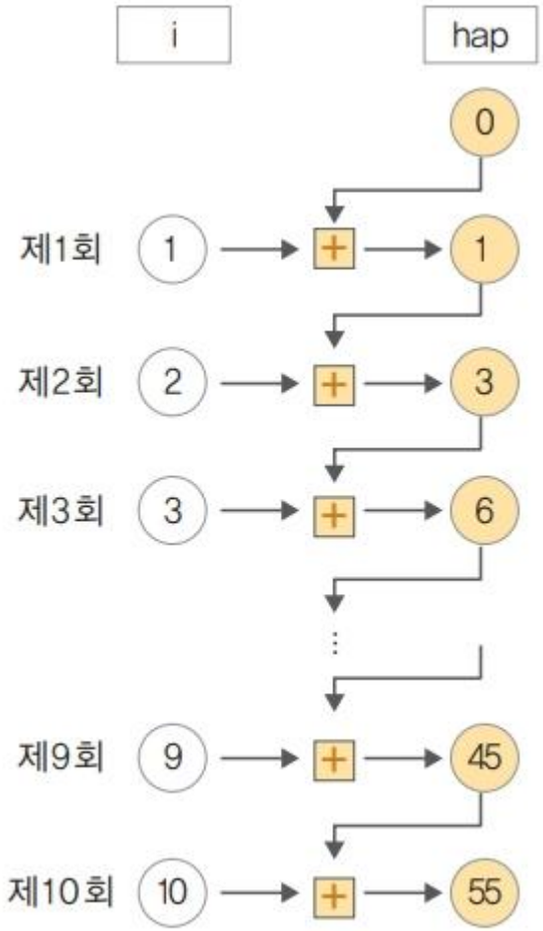


그림 6-4 변수 i와 hap의 변화

for문의 개념과 활용 (13/15)

- **for문의 활용 : 500 ~ 1000 까지의 숫자들 중 홀수들의 합 계산**

응용 6-9 for문을 활용하여 합계 구하기 3 6-9.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int hap = 0;
06    int i;
07
08    for( __1__; i <= 1000; __2__ ) {          ----- i를 501부터 2씩 증가시킨다.
09       hap = hap + i;
10    }
11
12    printf(" 500에서 1000까지의 홀수의 합: %d \n", hap);
13 }
```

정답_ 1 i=501 2 i=i+2 또는 i+=2

for문의 개념과 활용 (14/15)

- **for문의 활용 : 1부터 사용자 입력 숫자까지의 합을 계산**

기본 6-10 for문을 활용하여 합계 구하기 4 6-10.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int hap=0;          ----- 합계를 누적할 변수를 선언하고 0으로 초기화한다.
06    int i;              ----- 1씩 증가시킬 변수이다.
07    int num;            ----- 입력받을 최종값이다.
08
09    printf(" 값 입력 : ");
10    scanf("%d", &num);  ----- 최종값을 입력한다.
11
12    for( i=1; i <= num; i++ ) {   ----- 1부터 최종값까지 1씩 증가시키며 반복한다.
13       hap = hap + i;
14    }
15
16    printf(" 1에서 %d까지의 합: %d \n", num, hap);
17 }
```

for문의 개념과 활용 (15/15)

- **for문의 활용 : 시작값, 조건값, 증가값을 입력받기**

응용 6-11 for문을 활용하여 합계 구하기 5 6-11.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int hap=0;
06    int i;
07    int num1, num2, num3;          ---- 입력받을 변수 3개를 선언한다.
08
09    printf(" 시작값, 끝값, 증가값 입력 : ");
10    scanf("%d %d %d", &num1, &num2, &num3);   ---- 공백 문자로 구분해서 3개의 수를
11                                                 입력받는다.
12    for( __1__ ) {                 ---- 시작값은 num1, 끝값은 num2, 증가값은
13       hap = hap + i;                    num3이다.
14    }
15
16    printf(" %d에서 %d까지 %d씩 증가한 값의 합: %d \n", num1, num2, num3, hap);
17 }
```

정답_ 1 i=num1; i <= num2; i=i+num3

03
중첩 for문

중첩 for문 개념 (1/8)

- **중첩 for문은 for문 내부에 또 다른 for문이 들어 있는 형태**

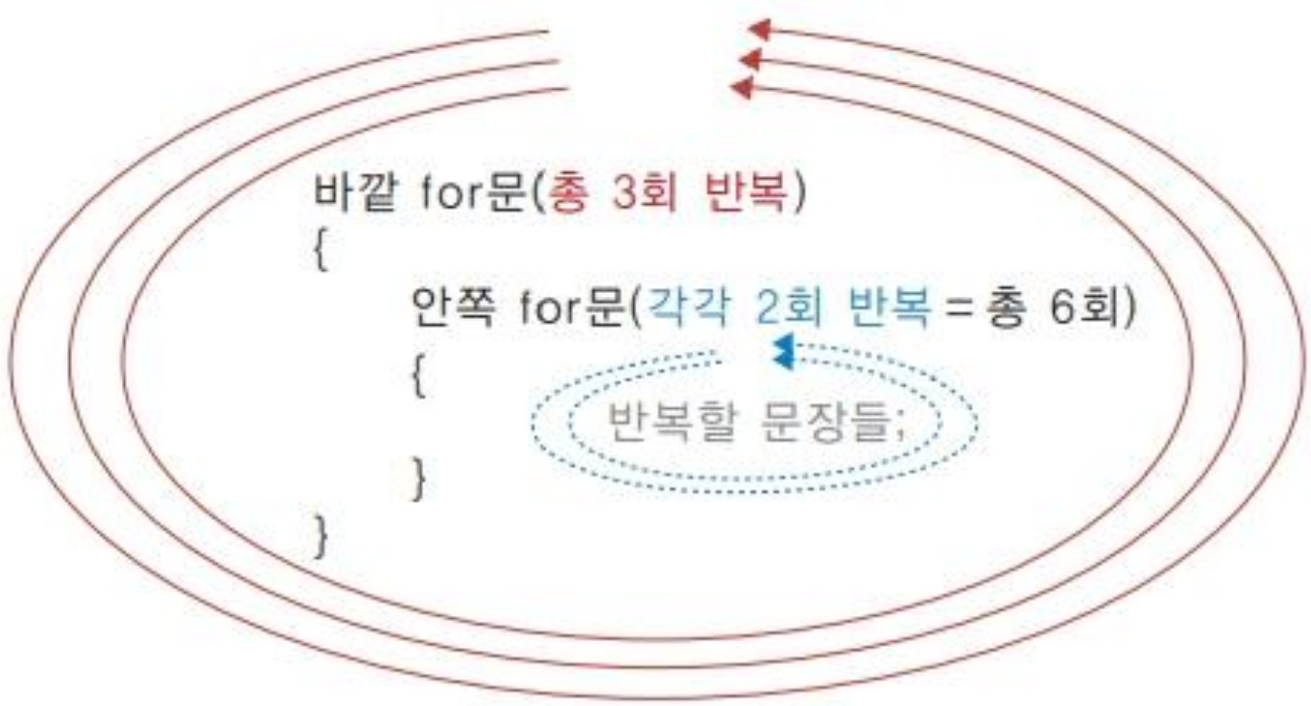


그림 6-5 중첩 for문의 동작 개념

중첩 for문 개념 (2/8)

- **중첩 for문 작동 순서**

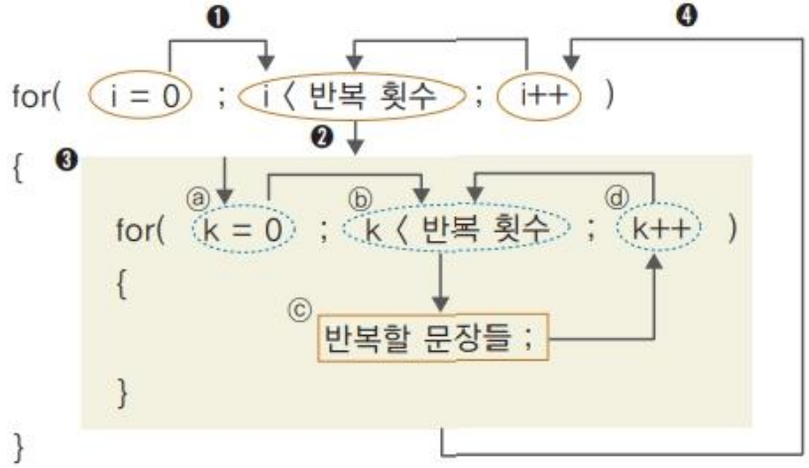


그림 6-6 중첩 for문의 작동 방식

✓ 바깥 for문의 반복 횟수를 3, 안쪽 for문의 반복 횟수를 2라고 가정

❶ → ❷ → ❸ → (ⓐ → ⓑ → ⓒ → ⓓ → ⓑ → ⓒ → ⓓ → ⓑ → 안쪽 for문을 빠져나감) → ❹ → ❷ → ❸ → (ⓐ → ⓑ → ⓒ → ⓓ → ⓑ → ⓒ → ⓓ → ⓑ → 안쪽 for문을 빠져나감) → ❹ → ❷ → ❸ → (ⓐ → ⓑ → ⓒ → ⓓ → ⓑ → ⓒ → ⓓ → ⓑ → 안쪽 for문을 빠져나감) → ❹ → ❷ → 바깥 for문을 빠져나감

중첩 for문 개념 (3/8)

기본 6-13 중첩 for문 사용 예 1 6-13.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i, k;                          ----- 반복할 변수 i, k를 선언한다.
06
07    for( i=0; i < 3; i++ )             ----- 바깥 for문을 세 번 반복한다.
08    {
09       for( k=0; k < 2; k++ )          ----- 안쪽 for문을 두 번 반복한다.
10       {
11          printf("중첩 for문입니다. (i값: %d, k값: %d)\n", i, k);
12       }                                     └---- i와 k 값을 총 여섯(3×2) 번 출력한다.
13    }
14 }
```

중첩 for문 개념 (4/8)

- **기본 6-13 복기**

❶ 외부 for문 제1회: 7행의 i를 0으로 초기화하면 i < 3이 참이므로 바깥 for문 수행
  내부 for문 제1회: 9행의 k를 0으로 초기화하면 k < 2가 참이므로 안쪽 for문 수행
    11행의 printf( )를 실행해서 '중첩 for문입니다.' 출력
  내부 for문 제2회: 9행의 k++로 k를 1로 증가시키면 k < 2가 참이므로 안쪽 for문 수행
    11행의 printf( )를 실행해서 '중첩 for문입니다.' 출력
  내부 for문 제3회: 9행의 k++로 k를 2로 증가시키면 k < 2가 거짓이므로 안쪽 for문 종료

❷ 외부 for문 제2회: 7행의 i++로 i를 1로 증가시키면 i < 3이 참이므로 바깥 for문 수행
  내부 for문 제1회: 9행의 k를 0으로 초기화하면 k < 2가 참이므로 안쪽 for문 수행
    11행의 printf( )를 실행해서 '중첩 for문입니다.' 출력
  내부 for문 제2회: 9행의 k++로 k를 1로 증가시키면 k < 2가 참이므로 안쪽 for문 수행
    11행의 printf( )를 실행해서 '중첩 for문입니다.' 출력
  내부 for문 제3회: 9행의 k++로 k를 2로 증가시키면 k < 2가 거짓이므로 안쪽 for문 종료

❸ 외부 for문 제3회: 7행의 i++로 i를 2로 증가시키면 i < 3이 참이므로 바깥 for문 수행
  내부 for문 제1회: 9행의 k를 0으로 초기화하면 k < 2가 참이므로 안쪽 for문 수행
    11행의 printf( )를 실행해서 '중첩 for문입니다.' 출력
  내부 for문 제2회: 9행의 k++로 k를 1로 증가시키면 k < 2가 참이므로 안쪽 for문 수행
    11행의 printf( )를 실행해서 '중첩 for문입니다.' 출력
  내부 for문 제3회: 9행의 k++로 k를 2로 증가시키면 k < 2가 거짓이므로 안쪽 for문 종료

❹ 외부 for문 제4회: 7행의 i++로 i를 3으로 증가시키면 i < 3이 거짓이므로 바깥 for문 종료

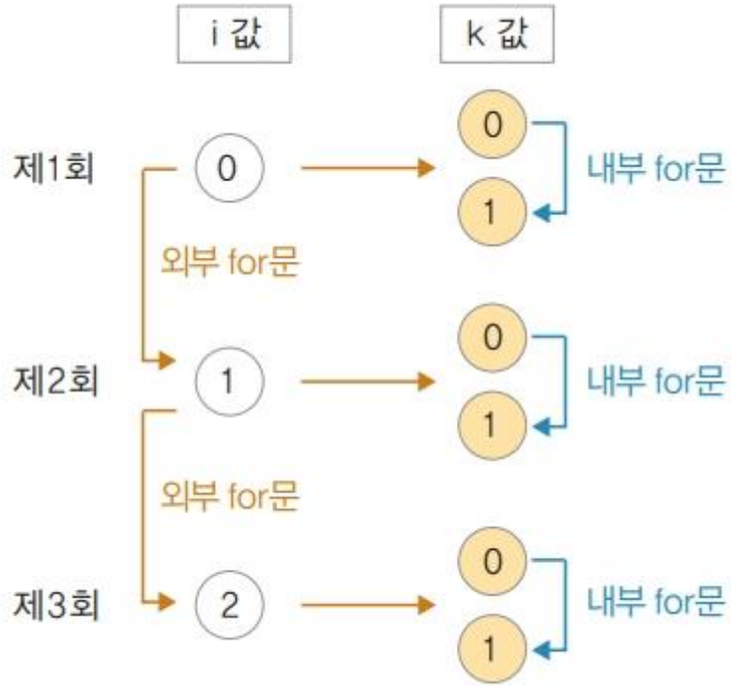


그림 6-7 중첩 for문에서 i 값과 k 값의 변화

중첩 for문 개념 (5/8)

- **응용 6-14 설계 : 구구단 2단부터 9단까지 출력**
  - ✓ for문의 조건식에 사용되는 변수 i와 k를 어떻게 설정해야 할까?

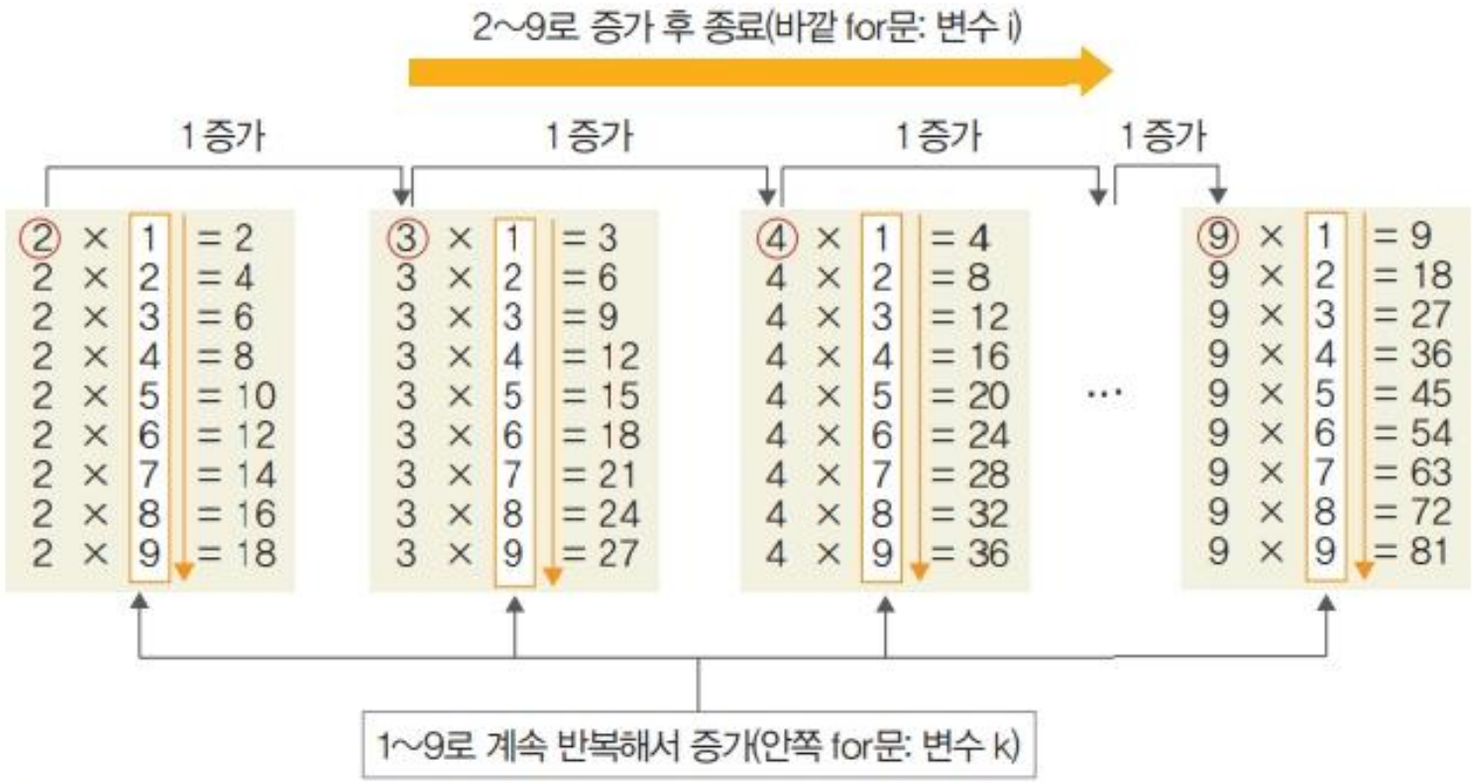


그림 6-8 구구단에서 변수 i와 k의 추출

중첩 for문 개념 (6/8)

응용 6-14 중첩 for문 사용 예 2

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i, k;
06
07    for( i = 2; i <= 9; i ++ )          ------ 2~9단까지 반복한다.
08    {
09       for( __1__ )                     ------ 각 단의 뒷자리는 1~9를 반복한다.
10       {
11          printf(" %d X %d =  %d \n", i, k, i*k);     ----- 구구단을 출력한다.
12       }
13       printf("\n");                    ----- 각 단이 끝나면 한 줄을 띄운다.
14    }
15 }
```

k = 1; k <= 9; k++

정답_ 1 k = 1; k <= 9; k++

실행 결과

```
2 X 1 =  2
2 X 2 =  4
2 X 3 =  6
2 X 4 =  8
2 X 5 =  10
2 X 6 =  12
2 X 7 =  14
2 X 8 =  16
2 X 9 =  18

3 X 1 =  3
3 X 2 =  6
   ⋮
   ⋮
```

```
   ⋮
   ⋮
8 X 7 =  56
8 X 8 =  64
8 X 9 =  72

9 X 1 =  9
9 X 2 =  18
9 X 3 =  27
9 X 4 =  36
9 X 5 =  45
9 X 6 =  54
9 X 7 =  63
9 X 8 =  72
9 X 9 =  81
```

중첩 for문 개념 (7/8)

- **구구단을 다음과 같은 형태로 출력하려면?**
  - ✓ for문의 조건식에 사용되는 변수 i와 k를 어떻게 바꾸어야 할까?

그림 6-9 구구단에서 변수 i와 k의 추출(단, 가로 먼저 출력)

중첩 for문 개념 (8/8)

응용 6-15 중첩 for문 사용 예 3 6-15.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i, k;
06
07    for( i = 1; i <= 9; i ++ )          ----- 각 단의 뒷자리는 1~9를 반복한다.
08    {
09       for( k = 2; k <= 9; k ++ )       ----- 2~9단까지 반복한다.
10       {
11          printf("%2dX%2d=%2d", __1__ );  ----- 각 단별로 한 줄씩 출력한다.
12       }
13       printf("\n");                    ----- 각 단의 한 줄을 출력한 후 다음 줄로 넘긴다.
14    }
15 }
```

k, i, k*i

정답_ 1 k, i, k*i

04
다양한 for문의 형태

여러 개의 초깃값과 증감식을 사용하는 for문

- **for문에 들어가는 초깃값, 증감식이 꼭 하나일 필요는 없음**
  - ✓ 단, 여러 개를 초기화할 때는 콤마(,)로 각 각을 구분
  - ✓ for(초깃값 1, 초깃값 2; 조건식; 증감식 1, 증감식 2)
  - ✓ 조건식은?

기본 6-16 for문의 다양한 활용 예 1 6-16.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i, k;                          ----- 반복할 변수 i, k를 선언한다.
06
07    for( i = 1,  k = 1; i <= 9; i++, k++ )   ----- 초깃값과 증감식이 2개이다.
08       printf(" %d X %d = %d \n", i, k, i*k);
09 }
```

실행 결과

```
1 X 1 = 1
2 X 2 = 4
3 X 3 = 9
4 X 4 = 16
5 X 5 = 25
6 X 6 = 36
7 X 7 = 49
8 X 8 = 64
9 X 9 = 81
```

초깃값과 증감식이 없는 for문 (1/5)

❶ 기본 형식

```
int i;
for( i = 0 ; i < 10 ; i ++ )
{
   printf("%d \n", i) ;
}
```

❷ 초깃값 빼기

```
int i;
i = 0;
for( ___ ; i < 10 ; i ++ )
{
   printf("%d \n", i) ;
}
```

❸ 초깃값과 증감식 빼기

```
int i;
i = 0;
for( ___ ; i < 10 ; ___ )
{
   printf("%d \n", i) ;
   i ++ ;
}
```

- 0~9를 출력하는 기본 형식은 ❶
- 이때 ❷와 같이 초깃값인 i=0을 for문 밖으로 빼서 사용할 수도 있음
- ❸과 같이 초깃값과 증감식을 모두 빼서 사용할 수도 있음

초깃값과 증감식이 없는 for문 (2/5)

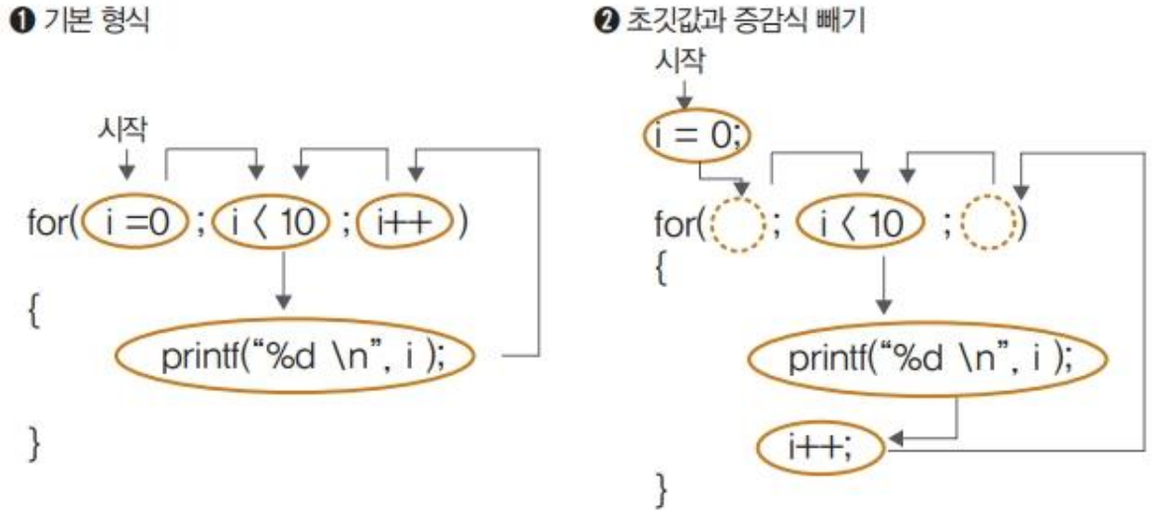


그림 6-10 두 소스의 비교

- **왼쪽(❶)과 오른쪽(❷) 소스의 실행 순서를 살펴보면 빈칸은 없는 것과 마찬가지이므로 결국 동일한 순서로 작동**

❶ 시작 → i=0 → i < 10 → printf( ) → i++ → i < 10 → printf( ) → i++ → ···

❷ 시작 → i=0 → 빈칸 → i < 10 → printf( ) → i++ → 빈칸 → i < 10 → printf() → i++ → ···

초깃값과 증감식이 없는 for문 (3/5)

기본 6-17 for문의 다양한 활용 예 2 — 6-17.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int i;
06    i = 0;
07    for(   ;   ;   )                  ----- 초깃값, 조건식, 증감식이 아무것도 없다.
08    {
09       printf("%d  \n", i);
10       i ++;
11    }
12 }
```

실행 결과

```
0
1
2
3
…
…
1000
1001
1002
1003
…
…
…
무한반복
```

초깃값과 증감식이 없는 for문 (4/5)

응용 6-18 for문의 다양한 활용 예 3 6-18.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int a, b;
06
07    __1__( ; ; )                          ----- 무한 루프가 발생한다.
08    {
09       printf("더할 두 수 입력 (멈추려면 Ctrl+C) : ");
10       scanf("%d %d", &a, &b);            ----- 두 값을 입력받는다.
11
12       printf("%d + %d = %d \n", a, b, a+b);   ----- 더하기 결과를 출력한다.
13    }
14 }
```

정답_1 for

초깃값과 증감식이 없는 for문 (5/5)

실행 결과

```
더할 두 수 입력 (멈추려면 Ctrl+C) : 11  33
11 + 33 = 44
더할 두 수 입력 (멈추려면 Ctrl+C) : 22  755
22 + 755 = 777
더할 두 수 입력 (멈추려면 Ctrl+C) :
```

- 7행은 for문 내부의 블록을 무한 루프에 빠지게 하는 구문
- 9행에서 설명문을 출력하고 10 행에서 두 수를 공백으로 구분해서 입력하면 12행에서 계산 결과가 출력
- Ctrl+C를 누를 때까지 반복

예 제

[예제 01] 구구단 프로그램

- **사용자로부터 단수를 입력 받아 결과를 출력**

기본 6-12 for문을 사용한 구구단 프로그램 6-12.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int i;
06    int dan;                    ----- 계산한 단을 입력받을 변수를 선언한다.
07
08    printf(" 몇 단 ? ");
09    scanf("%d", &dan);          ----- 계산할 단을 입력받는다.
10
11
12                                ----- 1부터 9까지 반복하며 입력한 단을
13                                      출력한다.
14 }
```

실행 결과

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

[예제 02] switch case문을 사용한 간단한 계산기

예제 설명 수식을 띄어쓰기로 한 줄에 입력받고 switch~case문을 활용하여 두 수의 +, –, *, /, % 연산을 수행하는 프로그램이다.

실행 결과

```
수식을 한 줄로 띄어쓰기로 입력하세요 : 99  -  22
99 - 22 = 77 입니다.
```

```
수식을 한 줄로 띄어쓰기로 입력하세요 : 55  #  99
연산자를 잘못 입력했습니다.
```

```
scanf("%d %c %d", &a, &ch, &b);
```

```
case '+':
  printf("%d + %d = %d 입니다. \n", a, b, a+b);
  break;
```

[예제 03] 구구단 출력 프로그램

- **중첩 for문을 사용하고, 제목도 출력하는 구구단 프로그램**

예제 설명 중첩 for문을 사용하여 제목과 구구단을 출력하는 프로그램이다.

몇단까지 출력할 지 사용자로부터 입력을 받자!

실행 결과

반복문을 이용하여 제목을 출력

```
#제2단# #제3단# #제4단# #제5단# #제6단# #제7단# #제8단# #제9단#

2X 1= 2 3X 1= 3 4X 1= 4 5X 1= 5 6X 1= 6 7X 1= 7 8X 1= 8 9X 1= 9
2X 2= 4 3X 2= 6 4X 2= 8 5X 2=10 6X 2=12 7X 2=14 8X 2=16 9X 2=18
2X 3= 6 3X 3= 9 4X 3=12 5X 3=15 6X 3=18 7X 3=21 8X 3=24 9X 3=27
2X 4= 8 3X 4=12 4X 4=16 5X 4=20 6X 4=24 7X 4=28 8X 4=32 9X 4=36
2X 5=10 3X 5=15 4X 5=20 5X 5=25 6X 5=30 7X 5=35 8X 5=40 9X 5=45
2X 6=12 3X 6=18 4X 6=24 5X 6=30 6X 6=36 7X 6=42 8X 6=48 9X 6=54
2X 7=14 3X 7=21 4X 7=28 5X 7=35 6X 7=42 7X 7=49 8X 7=56 9X 7=63
2X 8=16 3X 8=24 4X 8=32 5X 8=40 6X 8=48 7X 8=56 8X 8=64 9X 8=72
2X 9=18 3X 9=27 4X 9=36 5X 9=45 6X 9=54 7X 9=63 8X 9=72 9X 9=81
```

[예제 04] 아스키코드표 출력 프로그램

예제 설명 for문과 if문을 사용하여 아스키코드의 0~127을 10진수, 16진수, 문자로 출력하는 프로그램이다.

if문을 사용하여
16행마다 다음의 제목줄 출력

16진수는 %x,
문자는 %c로 출력

실행 결과

```
------------------
10진수  16진수  문자
------------------
   0     0
   1     1     ┌
   2     2     ┐
   3     3     └
   4     4     ┘
   5     5     |
   6     6     ─
   7     7
         ⋮
         ⋮
```

```
         ⋮
 116     74    t
 117     75    u
 118     76    v
 119     77    w
 120     78    x
 121     79    y
 122     7a    z
 123     7b    {
 124     7c    ¦
 125     7d    }
 126     7e    ~
 127     7f
```

Q & A