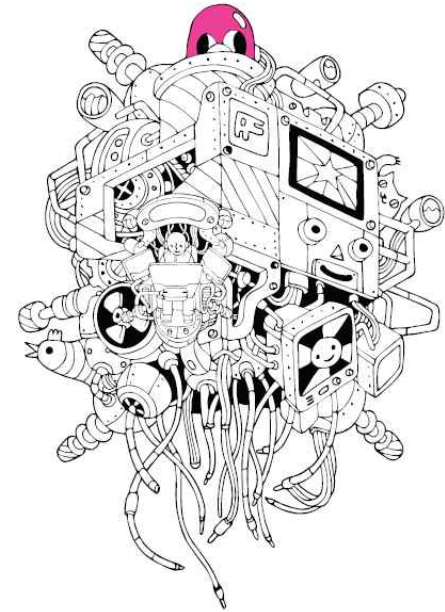


C 프로그래밍



반복문

프로그램 Flow Control의 종류

- ▶ 순차 구조

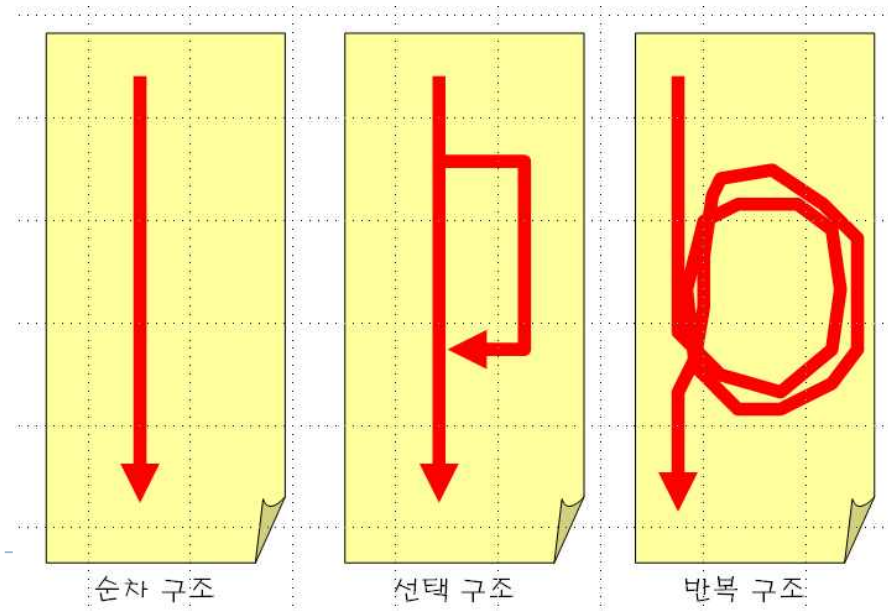
- ▶ 차례대로 실행

- ▶ 선택 구조

- ▶ 조건을 검사하여 여러 개의 실행 경로 중에서 하나를 선택

- ▶ 반복 구조

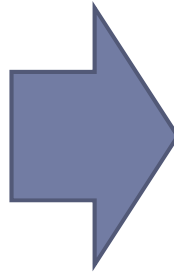
- ▶ 조건이 만족될 때까지 반복



반복문을 사용하는 이유

▶ 1부터 1000까지 화면에 프린트 하기

```
void main(void)
{
    // -----
    printf("%d \n", 1);
    printf("%d \n", 2);
    ...
    printf("%d \n", 999);
    printf("%d \n", 1000);
}
```



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

    for(int i=1;i<=1000;i++)
        printf("%d \n", i);
}
```



반복문의 종류

▶ while, for, do-while

```
while( 조건식 )  
{  
    문장;  
    문장;...  
}
```

```
for ( 초기화; 조건식; 증감식 )  
{  
    문장;  
}
```

```
do {  
    문장  
} while(조건)
```



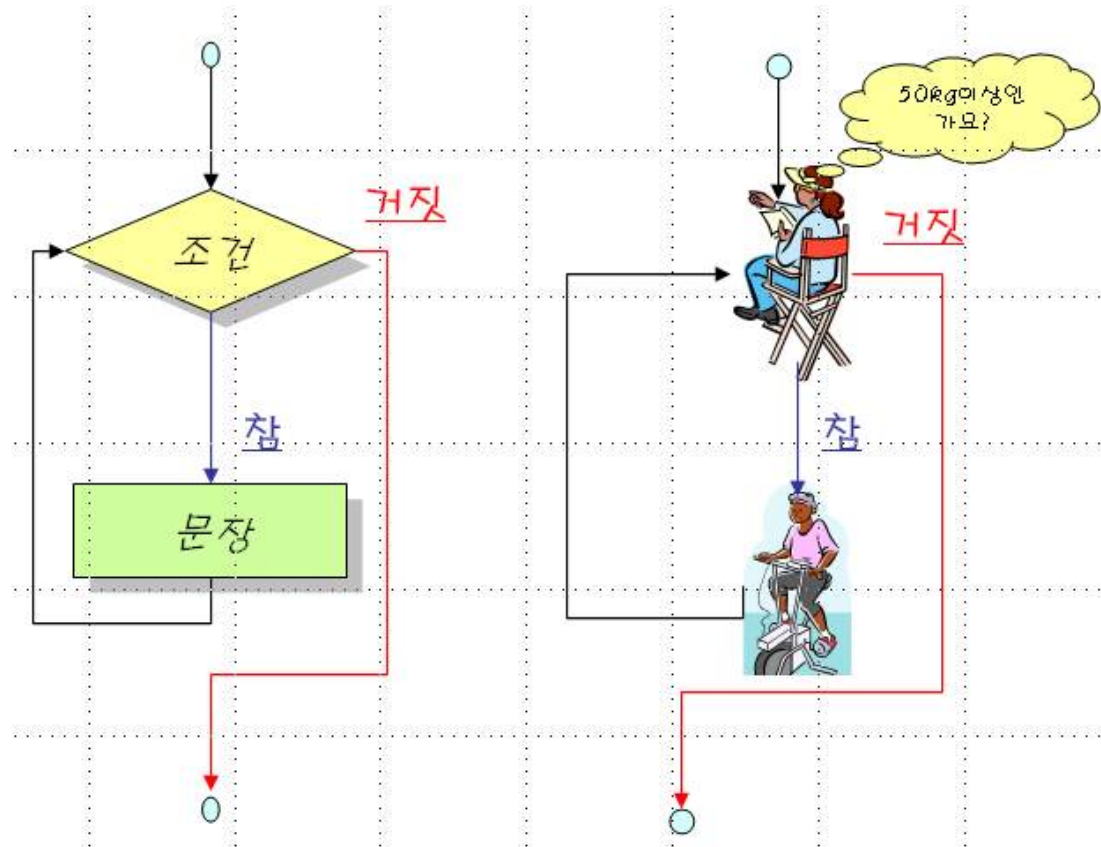


Chapter 07-1. while문에 의한 문장의 반복

반복문의 종류

▶ While() {}

```
while( 조건식 )  
{  
    문장;  
    문장;...  
}
```



반복문의 이해와 while문

- 반복문이란

하나 이상의 문장을 두 번 이상 반복 실행하기 위해서 구성하는 문장

- 반복문의 종류

while, do~while, for

반복의 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++;
```

```
int main(void)
```

```
{
```

```
    int num=0;
```

while 반복문

```
    while(num<5)
```

```
    {
```

```
        printf("Hello world! %d \n", num);
```

```
        num++;
```

중괄호 내부 반복영역

```
    }
```

```
    return 0;
```

```
}
```

반복의 목적이 되는 대상

변수 num은 반복의 횟수를 조절하기 위한 것!

```
Hello world! 0
```

```
Hello world! 1
```

```
Hello world! 2
```

```
Hello world! 3
```

```
Hello world! 4
```

실행결과

반복문 안에서도 들여쓰기 합니다.

들여쓰기를 하지 않은 것

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

들여쓰기를 한 것

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

들여쓰기를 한 것과 하지 않은 것의 차이가 쉽게 눈에 들어온다!

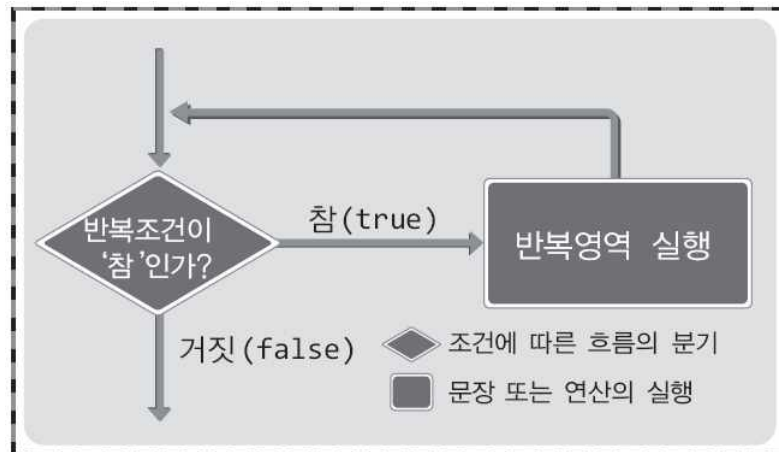


while문의 구성과 실행흐름의 세세한 관찰

```
int main(void)
{
    int num=0;
    while(num<3)  // 3회 반복
    {
        printf("Hello world! %d \n", num);
        num++;
    }
    . . . .
}
```



반복의 과정은?



flow chart 기준에서의 while문

예제 : 구구단의 출력

```
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++;
    }
    return 0;
}
```

```
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
```

실행결과

구구단은 반복문을 이해하는데 사용되는 대표적인 예제이다.
이후에 반복문의 중첩에서는 구구단 전체를 출력하는 예제를 접한다.



무한루프의 구성

```
while( 1 )
{
    printf("%d×%d=%d \n", dan, num, dan*num);
    num++;
}
```

숫자 1은 '참'을 의미하므로 반복문의 조건은 계속해서 '참'이 된다.

이렇듯 반복문의 탈출조건이 성립하지 않는 경우 무한루프를 형성한다고 한다.

이러한 무한루프는 실수로 만들어지는 경우도 있지만, break문과 함께 유용하게 사용되기도 한다.



while문의 중첩

while문 안에 while문이 존재하는 상태를 의미한다. 아래의 예제에서는 while문을 중첩시켜서 구구단 전체를 출력한다. 이 예제를 통해서 중첩된 while문의 코드 흐름을 이해하자.

```
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
    }

    return 0;
}
```

바깥쪽 while문

안쪽 while문



예제



// while 문을 이용한 제곱값 출력 프로그램

#include <stdio.h>

```
int main(void)
{
```

```
    int n;
```

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

```
    printf("  n      n의 제곱 \n");
```

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

```
    n = 1;
```

```
    while (n <= 10)
```

```
    {
```

```
        printf("%5d  %5d\n", n, n*n);
```

```
        n++;
```

```
    }
```

```
    return 0;
```

```
}
```



=====	
n	n의 제곱
=====	
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81
10	100

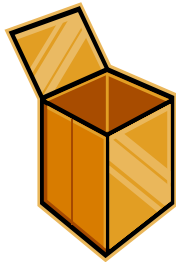
예제

▶ 1부터 n까지의 합을 계산하는 프로그램

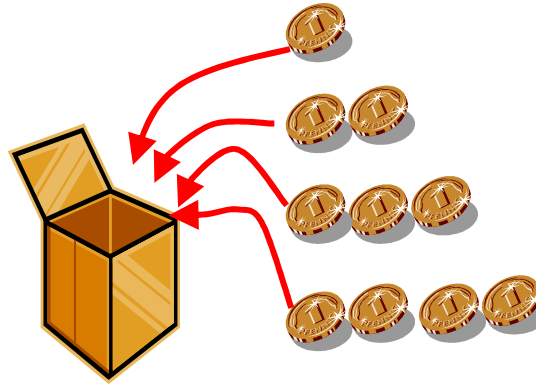
$$1 + 2 + 3 + \dots + n$$

- n이 무엇이 될지 모르는 경우라면 다음과 같이 계산

① 빈통을 준비한다.

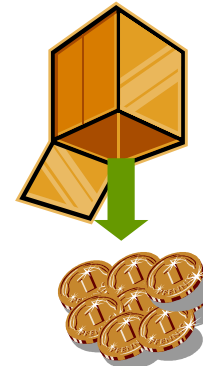


② 통에 1부터 n까지를 넣는다.



...

③ 통에 들어 있는 동전의 개수를 출력한다.



예제



```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int i, sum;
```

```
    i = 1;
```

```
    sum = 0;
```

```
    while (i <= 1000)
```

```
    {
```

```
        sum += i;
```

```
        i++;
```

```
    }
```

```
    printf("합은 %d입니다.\n", sum);
```

```
    return 0;
```

```
}
```



합은 500500입니다.



예제



```
// while 문을 이용한 합계 프로그램  
#include <stdio.h>
```

```
int main(void)  
{  
    int i, n, sum;  
  
    i = 0;                // 변수 초기화  
    sum = 0;              // 변수 초기화  
    while (i < 5)  
    {  
        printf("값을 입력하시오: ");  
        scanf("%d", &n);  
        sum = sum + n;    // sum += n;과 같다.  
        i++;  
    }  
    printf("합계는 %d입니다.\n", sum);  
  
    return 0;  
}
```



값을 입력하시오: 10
값을 입력하시오: 20
값을 입력하시오: 30
값을 입력하시오: 40
값을 입력하시오: 50
합계는 150입니다.

Advanced Programming!!

▶ 커서를 원하는 곳에 위치하기(gotoxy, clrscr)

```
#include <stdio.h>
#include <conio.h> // _getch();
#include <windows.h> // gotoxy()내 함수
#include <stdlib.h> // system("cls")

void gotoxy(int x, int y) // 그대로 복사해서 사용
{
    COORD Pos = {x - 1, y - 1};
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}

void main(void)
{
    gotoxy(2,4);
    printf("Hello~");
    gotoxy(40, 20);
    printf("Hello\n");
    _getch();

    system("cls");
    _getch();
}
```



Advanced Programming!!

▶ 한칸씩 밀어서 구구단 출력하기

```
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++;
    }
    return 0;
}
```

```
#include <stdio.h>
#include <conio.h> // _getch();
#include <windows.h> // gotoxy()내 함수
#include <stdlib.h> // system("cls")

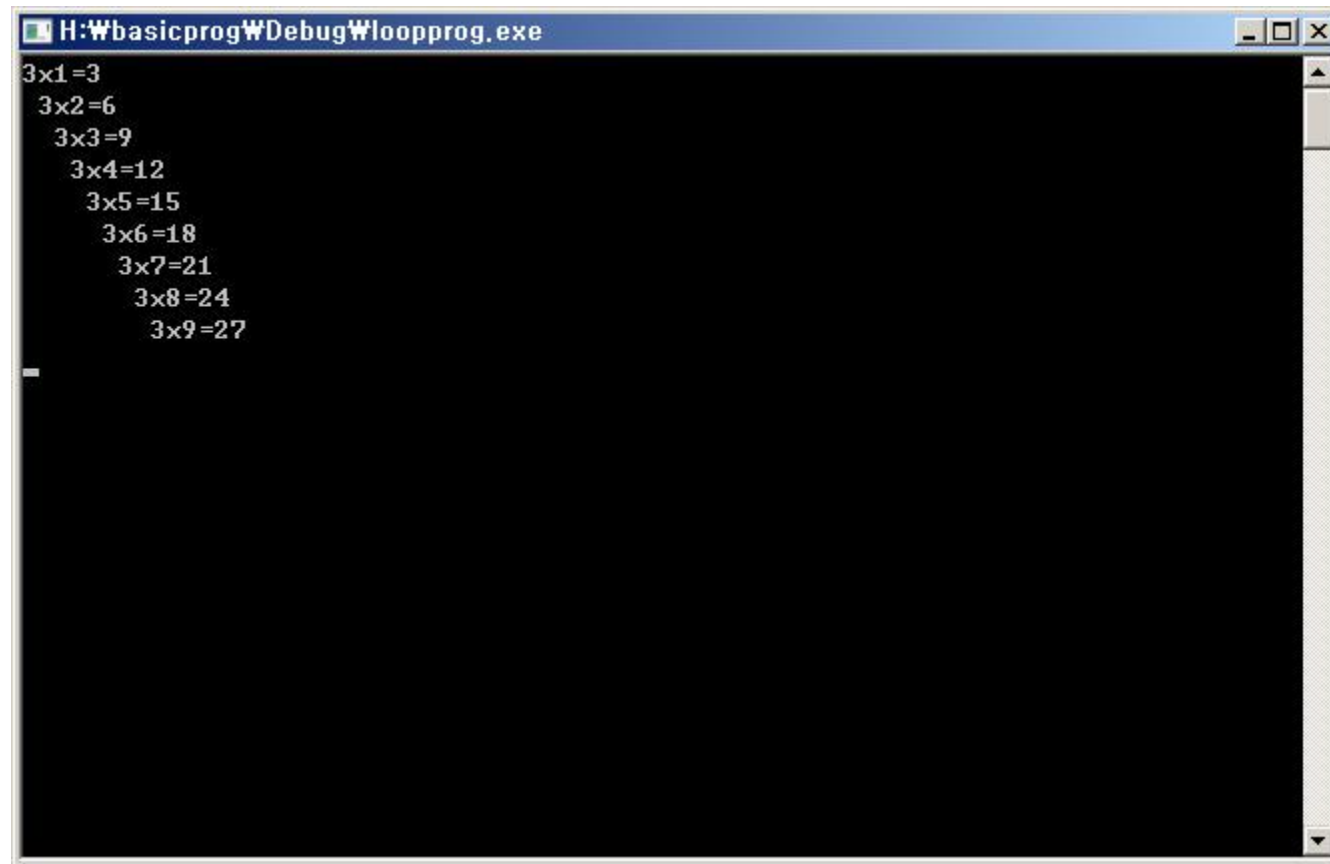
void gotoxy(int x, int y) // 그대로 복사해서 사용
{
    COORD Pos = {x - 1, y - 1};
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}

void main(void)
{
    int dan =0, num = 1;
    printf(" 몇 단? ");
    scanf("%d", &dan);

    system("cls");
    while(num<10)
    {
        gotoxy(num, num);
        printf("%dx%d=%d \n", dan, num, dan*num);
        num++;
    }
    _getch();
}
```

Advanced Programming!!

- ▶ 한칸씩 밀어서 구구단 출력하기



```
H:\basicprog\WDebug\Wloopprog.exe
3x1=3
 3x2=6
 3x3=9
 3x4=12
 3x5=15
 3x6=18
 3x7=21
 3x8=24
 3x9=27
```

The screenshot shows a Windows command prompt window with the title bar "H:\basicprog\WDebug\Wloopprog.exe". The window contains a text-based multiplication table for the number 3. The table is right-aligned, with each line representing a multiplication from 3x1 to 3x9. The text is as follows:

3x1=3
3x2=6
3x3=9
3x4=12
3x5=15
3x6=18
3x7=21
3x8=24
3x9=27

A small cursor is visible on the line following the last row of the table.

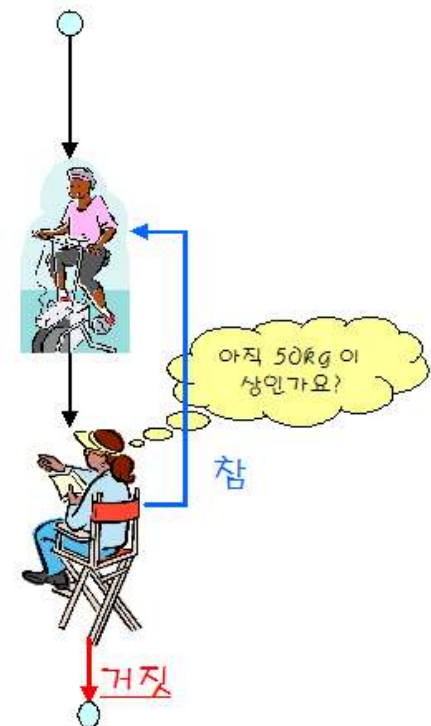
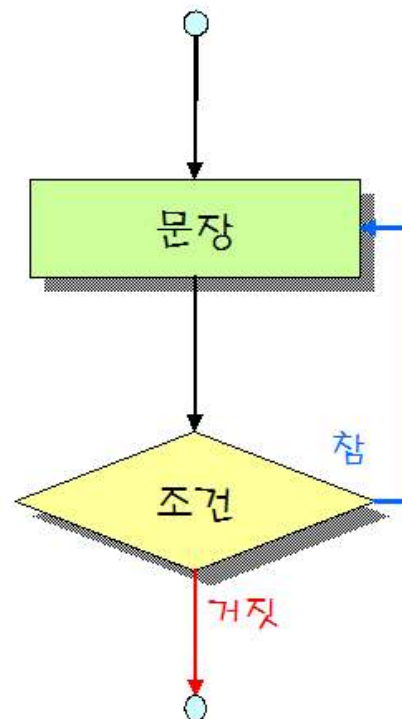


Chapter 07-2. do~while문에 의한 문장의 반복

반복문의 종류

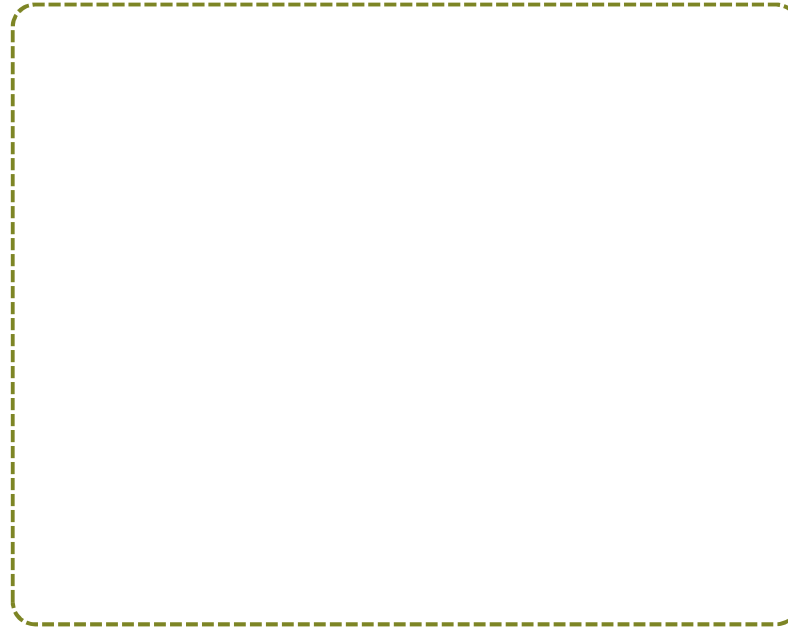
▶ do{ } while()

```
do {  
    문장  
} while(조건)
```



do~while문의 기본구성

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



반복의 과정은?

반복조건을 반복문의 마지막에 진행하는 형태이기 때문에
최소한 1회는 반복영역을 실행하게 된다. 이것이 while문과의 가장 큰 차이점이다.

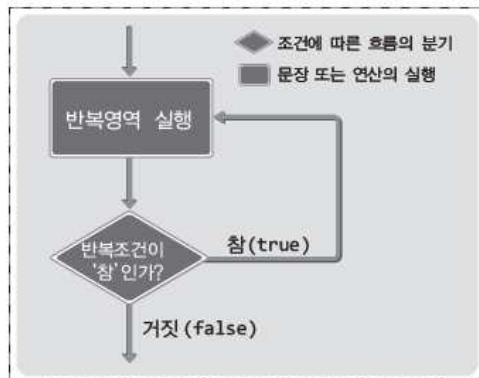


do~while문이 자연스러운 상황

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

↕ 동일한 횟수를 반복하는 반복문들

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



do~while문의 순서도

```
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);
    return 0;
}
```

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

실행결과

최소한 1회 이상 실행되어야 하는 반복문은 do~while문으로 구성하는 것이 자연스럽다.

예제



// do..while 문을 이용한 메뉴

#include <stdio.h>

int main(void)
{

int i = 0;

do

{

printf("1---새로만들기\n");

printf("2---파일열기\n");

printf("하나를 선택하시요.\n");

scanf("%d", &i);

} while(i < 1 || i > 2);

printf("선택된 메뉴=%d\n", i);

return 0;

}



1---새로만들기

2---파일열기

하나를 선택하시요.

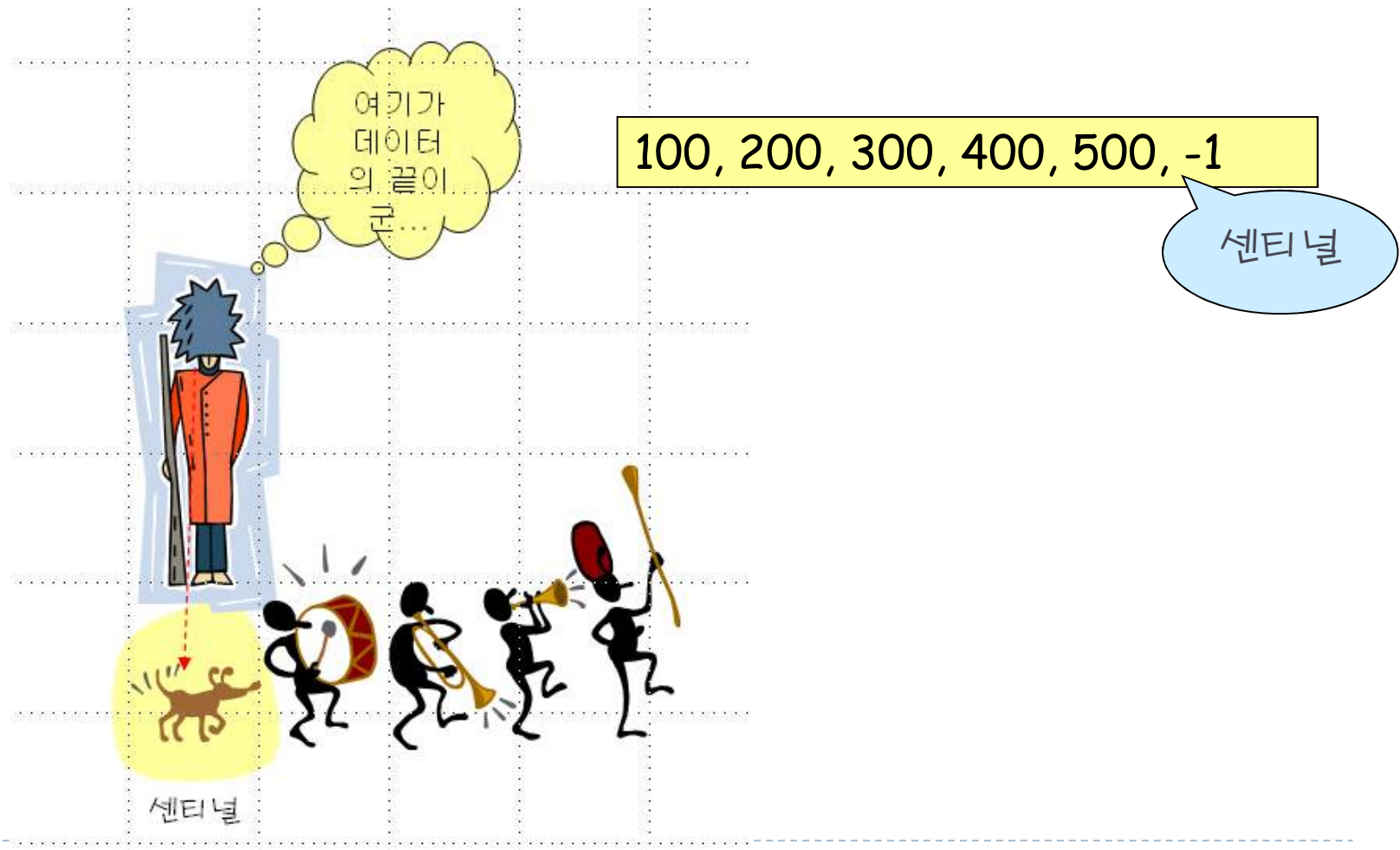
1

선택된 메뉴=1



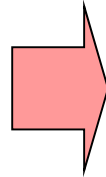
센티널(보초값의 이용)

- ▶ 센티널: 입력되는 데이터의 끝을 알리는 특수한 값



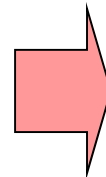
성적들의 평균을 구하는 문제

- 성적의 평균을 구한다.



1. 필요한 변수들을 초기화한다.
2. 성적을 입력받아서 합계를 구하고 성적의 개수를 센다.
3. 평균을 계산하고 화면에 출력한다.

1. 필요한 변수들을 초기화한다.

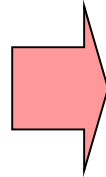


- (1) **sum**을 0으로 초기화한다.
- (2) **n**을 0으로 초기화한다.
- (3) **grade**를 0으로 초기화한다.



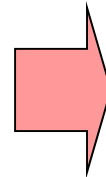
성적들의 평균을 구하는 문제

2. 성적을 입력받아서 합계를 구하고 성적의 개수를 센다.



while 성적이 0보다 작지 않으면
(1) 사용자로부터 성적을 읽어서 **grade**에 저장한다.
(2) **sum**에 이 점수를 누적한다.
(3) **n**을 하나 증가한다.

3. 평균을 계산하고 화면에 출력한다.



(1) **sum**을 **n**으로 나누어서 **average**에 저장한다.
(2) **average**를 화면에 출력한다.



센티넬 예제 1/2



```
// while 문을 이용한 성적의 평균 구하기 프로그램  
#include <stdio.h>
```

```
int main(void)  
{  
    int grade, n;  
    float sum, average;  
  
    // 필요한 변수들을 초기화한다.  
    n = 0;  
    sum = 0;  
    grade = 0;  
  
    printf("성적 입력을 종료하려면 음수를 입력하시오\n");
```

센티넬 예제 2/2

// 성적을 입력받아서 합계를 구하고 학생 수를 센다.

```
while (grade >= 0)
```

```
{
```

```
    printf("성적을 입력하시오: ");  
    scanf("%d", &grade);
```



```
    sum += grade;  
    n++;
```

```
}
```

```
sum = sum - grade; // 마지막 데이터를 제거한다.
```

```
n--; // 마지막 데이터를 제거한다.
```

```
// 평균을 계산하고 화면에 출력한다.
```

```
average = sum / n;
```

```
printf("성적의 평균은 %f입니다.\n", average);
```

```
return 0;
```

```
}
```

성적 입력을 종료하려면 음수를 입력하시오

성적을 입력하시오: 10

성적을 입력하시오: 20

성적을 입력하시오: 30

성적을 입력하시오: 40

성적을 입력하시오: 50

성적을 입력하시오: -1

성적의 평균은 30.000000입니다.

Advanced Programming

▶ 키보드의 키값을 프린트하기

```
#include <stdio.h>

int main(void)
{
    char ch = 0;

    do
    {
        printf( "<코드값을 알고 싶은 키를 누르시오>" );
        ch = _getch();
        printf( "\n 키 : %c, ASCII(10진수) : %d, (16진수): %x\n", ch, ch, ch );
    } while( ch!= '0' );

}
```



Advanced Programming

- ▶ 키보드의 키값을 프린트하기 : 출력결과
 - ▶ 일반 키보드키값은 1 byte(ASCII), 확장키값(2byte)

```
H:\Wbasicprog\WDebug\Wloopprog.exe
<코드값을 알고 싶은 키를 누르시오>
키 : a, ASCII<10진수> : 97, <16진수>: 61
<코드값을 알고 싶은 키를 누르시오>
키 : a, ASCII<10진수> : 97, <16진수>: 61
<코드값을 알고 싶은 키를 누르시오>
키 : e, ASCII<10진수> : 64, <16진수>: 40
<코드값을 알고 싶은 키를 누르시오>
키 : 2, ASCII<10진수> : 50, <16진수>: 32
<코드값을 알고 싶은 키를 누르시오>
키 : ? ASCII<10진수> : -32, <16진수>: ffffffff0
<코드값을 알고 싶은 키를 누르시오>
키 : H, ASCII<10진수> : 72, <16진수>: 48
<코드값을 알고 싶은 키를 누르시오>
키 : ? ASCII<10진수> : -32, <16진수>: ffffffff0
<코드값을 알고 싶은 키를 누르시오>
키 : M, ASCII<10진수> : 77, <16진수>: 4d
<코드값을 알고 싶은 키를 누르시오>
키 : ? ASCII<10진수> : -32, <16진수>: ffffffff0
<코드값을 알고 싶은 키를 누르시오>
키 : P, ASCII<10진수> : 80, <16진수>: 50
<코드값을 알고 싶은 키를 누르시오>
키 : ? ASCII<10진수> : -32, <16진수>: ffffffff0
<코드값을 알고 싶은 키를 누르시오>
키 : K, ASCII<10진수> : 75, <16진수>: 4b
<코드값을 알고 싶은 키를 누르시오>
```

확장키
ch==0
또는
ch==0xe0
(-32)

화살표
위,오른쪽,아래,왼쪽 키값

Advanced Programming

▶ 화살표 키를 이용하여 화면에 그림 그리기

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    const int MAX_X = 80; //1-80
```

```
    const int MAX_Y = 25; //1-24
```

```
    char key = 0;
```

```
    int x1=10, y1=5;
```

```
    system("cls");
```

```
    do
```

```
    {
```

```
        gotoxy(x1, y1);
```

```
        printf("A");
```

```
        key= _getch();
```

```
        switch( key )
```

```
        {
```

```
            case 72 : // 위로 화살표
```

```
                y1 = y1 -1;
```

```
                if ( y1<1)    y1 = 1;
```

```
                break;
```

```
            case 75 :
```

```
                x1 = x1-1;
```

```
                if(x1<1)    x1 = 1;
```

```
                break;
```

```
            case 77 :
```

```
                x1 = x1 + 1;
```

```
                if (x1 > MAX_X)    x1 = MAX_X;
```

```
                break;
```

```
            case 80 :
```

```
                y1 = y1 + 1;
```

```
                if (y1 > MAX_Y)    y1 = MAX_Y;
```

```
                break;
```

```
        }
```

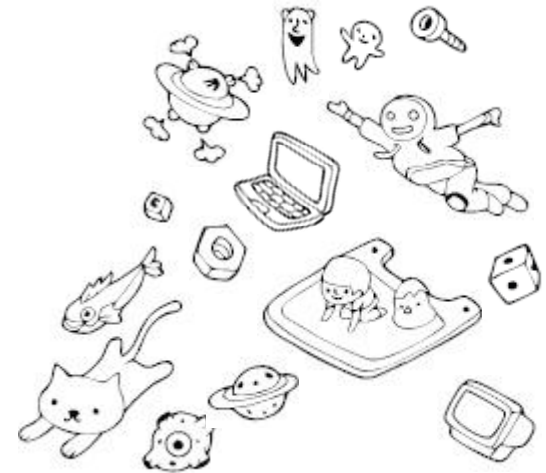
```
    } while( key!=27); // ESC key라면
```

```
}
```


Advanced Programming

- ▶ 화살표 키를 이용하여 화면에 그림 그리기





Chapter 07-3. for문에 의한 문장의 반복

반복문의 필수3요소

```
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
    }
    . . . .
}
```

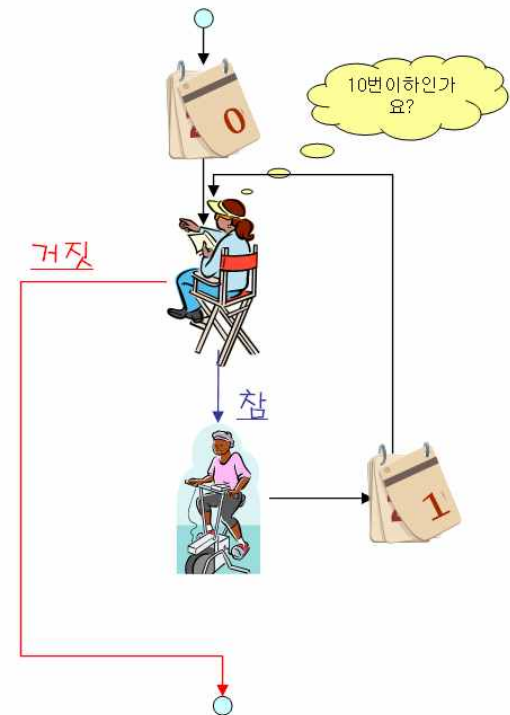
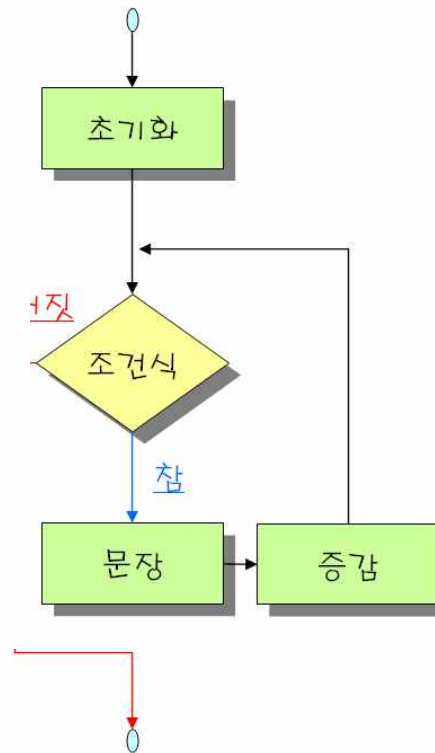
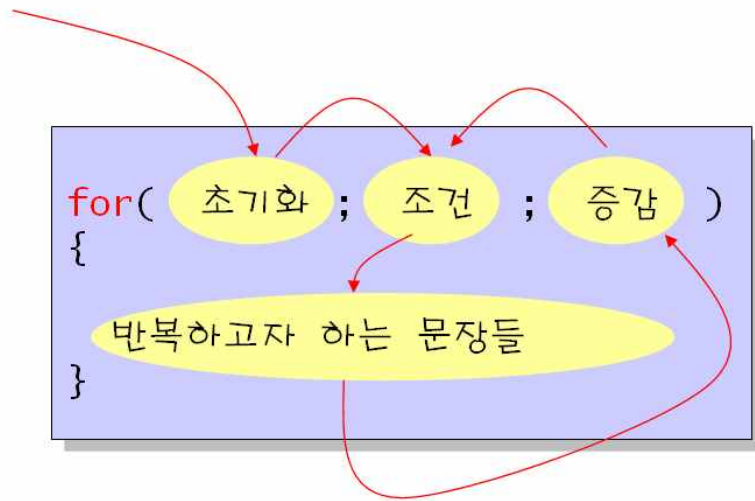
정해진 횟수의 반복을 위해서는 하나의 변수가 필요하다.
그 변수를 기반으로 하는 조건검사가 필요하다.
조건검사가 false가 되게 하기 위한 연산이 필요하다.

이 세 가지를 한 줄에 표시하도록
돕는 것이 for문이다.

위의 while문에서 보이듯이 반복문에 필요한 세 가지 요소가 여러 행에 걸쳐서 분산되어 있다.
따라서 반복의 횟수가 바로 인식 불가능하다.

for 반복문

▶ for(;;){}

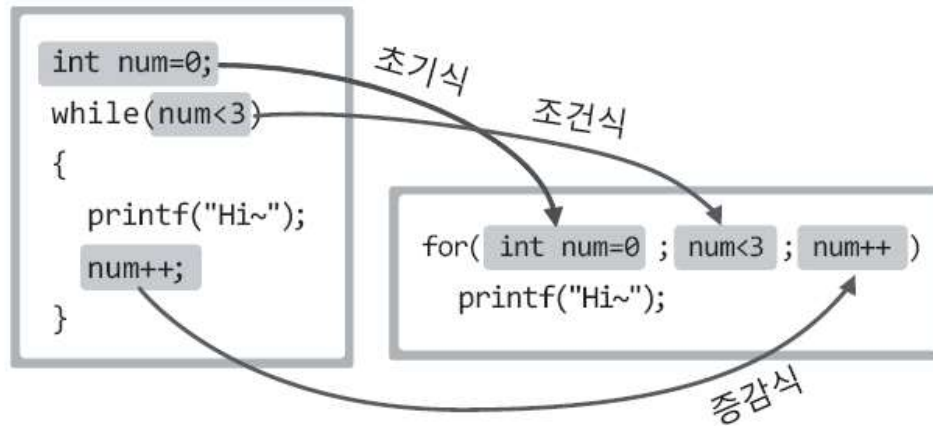


for 문의 예

```
int i;  
for(i = 0; i < 10; i++)  
    printf("Hello World!\n");
```



for문의 구조와 이해



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

```
int main(void)
{
    int num;
    for(num=0; num<3; num++)
        printf("Hi~");
    . . .
}
```

일부 컴파일러는 여전히 초기식에서의 변수 선언을 허용하지 않는다.
for문의 반복영역도 한 줄이면 중괄호 생략 가능!

for문의 흐름 이해

for문의 구성요소

- ✓ 초기식 본격적으로 반복을 시작하기에 앞서 딱 한번 실행된다.
- ✓ 조건식 매 반복의 시작에 앞서 실행되며, 그 결과를 기반으로 반복유무를 결정!
- ✓ 증감식 매 반복실행 후 마지막에 연산이 이뤄진다.

for문 흐름의 핵심

int num=0에 해당하는 초기화는 반복문의 시작에 앞서 딱 1회 진행!
num<3에 해당하는 조건의 검사는 매 반복문의 시작에 앞서 진행!
num++에 해당하는 증감연산은 반복영역을 실행한 후에 진행!

① 첫 번째 반복의 흐름
1 → 2 → 3 → 4 [num=1]

② 두 번째 반복의 흐름
2 → 3 → 4 [num=2]

③ 세 번째 반복의 흐름
2 → 3 → 4 [num=3]

④ 네 번째 반복의 흐름
2 [num=3] 따라서 탈출!

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

for문 기반의 다양한 예제

```
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);
    return 0;
}
```

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

실행결과

다양한 예제를 통해서 for문에 익숙해지자!

오른쪽 예제에서 보이듯이 불필요하다면, 초기식, 조건식, 증감식을 생략할 수 있다.
단 조건식을 생략하면 참으로 인식이 되어 무한루프를 형성하게 된다.

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

실행결과

```
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));
    return 0;
}
```


예제



```
// 반복을 이용한 정수합 프로그램
#include <stdio.h>

int main(void)
{
    int i, sum;

    sum = 0;
    for(i = 1; i <= 10; i++)
        sum += i;           // sum = sum + i;와 같음

    printf("1부터 10까지의 정수의 합 = %d\n", sum);

    return 0;
}
```



1부터 10까지의 정수의 합 = 55

예제



```
// 반복을 이용한 세제곱값구하기
#include <stdio.h>
```

```
int main(void)
{
    int i, n;

    printf("정수를 입력하시요:");
    scanf("%d", &n);

    printf("=====\n");
    printf(" i      i의 세제곱\n");
    printf("=====\n");
    for(i = 1; i <= n; i++)
        printf("%5d    %5d\n", i, i*i*i);

    return 0;
}
```



정수를 입력하시요:5

=====

i i의 세제곱

=====

1	1
2	8
3	27
4	64
5	125



예제



// 반복을 이용한 네모 그리기

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int i;
```

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

```
    for(i = 0; i < 5; i++)
```

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

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

```
    return 0;
```

```
}
```



```
*****  
*          *  
*          *  
*          *  
*          *  
*          *  
*****
```



예제



// 반복을 이용한 팩토리얼 구하기

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    long fact=1;
```

```
    int i, n;
```

```
    printf("정수를 입력하시요:");
```

```
    scanf("%d", &n);
```

```
    for(i = 1; i <= n; i++)
```

```
        fact = fact * i;
```

```
    printf("%d!은 %d입니다.\n",n,fact);
```

```
    return 0;
```

```
}
```



정수를 입력하시요: 10

10!은 3628800입니다.

실습 : 한글코드 출력하기

▶ KSC5601에서 완성형 한글코드 출력하기

▶ 2byte 코드 중에서

▶ b1: 0xb0-0xc8 b2: 0xa1-0xfe

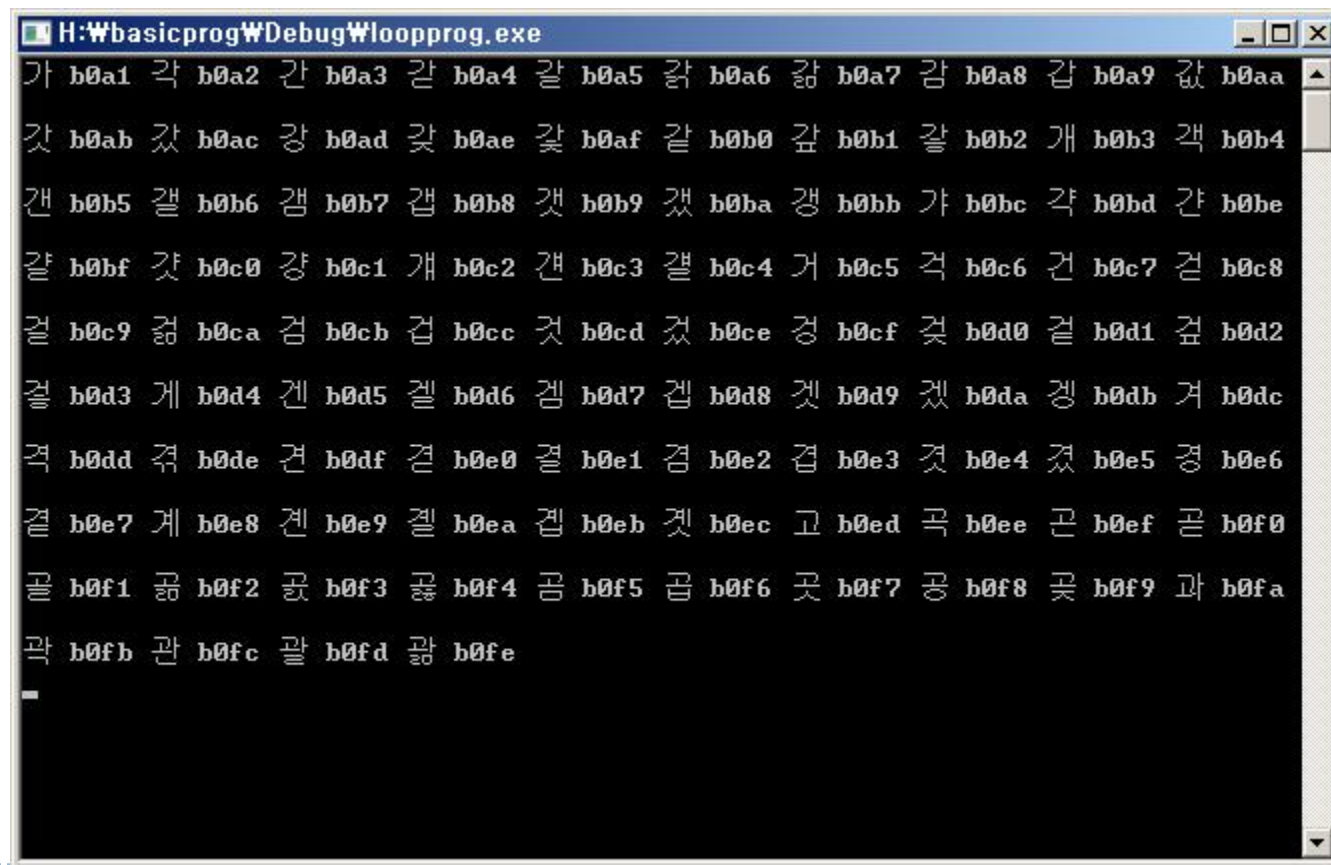
```
// 반복을 이용한 팩토리얼 구하기
#include <stdio.h>
#include <conio.h>

void main(void)
{
    unsigned char b1 = 0xb0;
    unsigned char b2 = 0x00;

    for( b2 = 0xa1; b2<= 0xfe; b2++ )
    {
        printf("%c%c %2x%2x ", b1, b2, b1, b2);
        if (b2%10 == 0)
            printf("\n");
    }
    printf("\n");
    _getch();
}
```

실습 : 한글코드 출력하기

▶ KSC5601에서 완성형 한글코드 출력하기



```
H:\Wbasicprog\WDebug\Wloopprog.exe
가 b0a1 각 b0a2 간 b0a3 갈 b0a4 갈 b0a5 감 b0a6 감 b0a7 감 b0a8 갑 b0a9 값 b0aa
갇 b0ab 갇 b0ac 갇 b0ad 갇 b0ae 갇 b0af 갈 b0b0 값 b0b1 갈 b0b2 개 b0b3 객 b0b4
갸 b0b5 갸 b0b6 갸 b0b7 갸 b0b8 갸 b0b9 갸 b0ba 갸 b0bb 가 b0bc 각 b0bd 간 b0be
갈 b0bf 갇 b0c0 갇 b0c1 개 b0c2 갸 b0c3 갸 b0c4 거 b0c5 격 b0c6 건 b0c7 겉 b0c8
겉 b0c9 겉 b0ca 겉 b0cb 겉 b0cc 겉 b0cd 겉 b0ce 겉 b0cf 겉 b0d0 겉 b0d1 겉 b0d2
겉 b0d3 게 b0d4 겉 b0d5 겉 b0d6 겉 b0d7 겉 b0d8 겉 b0d9 겉 b0da 겉 b0db 겨 b0dc
격 b0dd 겨 b0de 겉 b0df 겉 b0e0 겉 b0e1 겉 b0e2 겉 b0e3 겉 b0e4 겉 b0e5 겉 b0e6
겉 b0e7 게 b0e8 겉 b0e9 겉 b0ea 겉 b0eb 겉 b0ec 고 b0ed 곡 b0ee 곧 b0ef 곧 b0f0
곧 b0f1 곧 b0f2 곧 b0f3 곧 b0f4 곧 b0f5 곧 b0f6 곧 b0f7 곧 b0f8 곧 b0f9 과 b0fa
괘 b0fb 관 b0fc 괘 b0fd 괘 b0fe
```

for문의 중첩

9*9 단 전체를 출력하는 프로그램

```
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");
    }
    return 0;
}
```

for문의 중첩은 while, do~while문의 중첩과 다르지 않다.

구구단 전체를 출력하는 원편의 예제를 통해서 for문의 중첩을 이해하자.



다양한 증감수식의 형태

```
for (i = 10; i > 0; i-- )  
    printf("Hello World!\n");
```

백셈 사용

```
for (i = 0; i < 10; i += 2 )  
    printf("Hello World!\n");
```

2씩 증가

```
for (i = 1; i < 10; i *= 2 )  
    printf("Hello World!\n");
```

2를 곱한다.

```
for (i = 0; i < 100; i = (i * i) + 2 )  
    printf("Hello World!\n");
```

어떤 수식이라도 가능

```
for ( ; i < 100; i++ )  
    printf("Hello World!\n");
```

한부분이 없을 수도 있다.

```
for (i = 0, k = 0; i < 100; i++ )  
    printf("Hello World!\n");
```

2개 이상의 변수 초기화

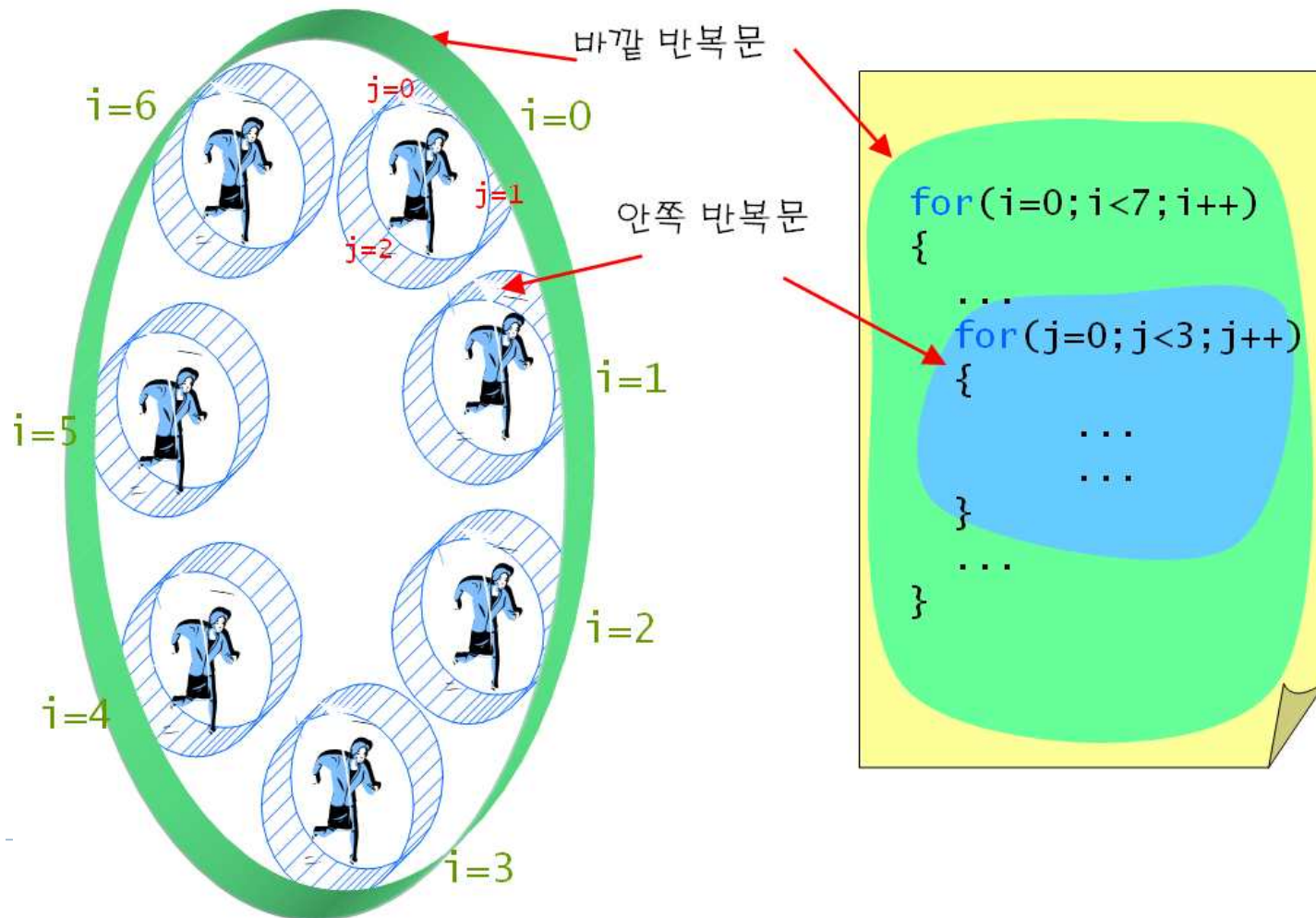
```
for (printf("반복시작"), i = 0; i < 100; i++ )  
    printf("Hello World!\n");
```

어떤 수식도 가능



중첩 반복문

- ▶ 중첩 반복문(nested loop): 반복문 안에 다른 반복문이 위치



예제

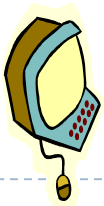


// 중첩 for 문을 이용하여 *기호를 사각형 모양으로 출력하는 프로그램
`#include <stdio.h>`

```
int main(void)
{
    int x, y;

    for(y = 0; y < 5; y++)
    {
        for(x = 0; x < 10; x++)
            printf("*");
        printf("\n");
    }

    return 0;
}
```



```
*****
*****
*****
*****
*****
```

예제



```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int x, y;
```

```
    for(y = 1; y <= 5; y++)
```

```
    {
```

```
        for(x = 0; x < y; x++)
```

```
            printf("*");
```

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

```
    }
```

```
    return 0;
```

```
}
```



```
*
```

```
**
```

```
***
```

```
****
```

```
*****
```

중간 점검

1. 다음 코드의 출력을 쓰시오.

```
for(i = 1; i < 6; i++)  
    for(j = 5; j >= 1; j--)  
        printf("%d 곱하기 %d은 %d\n", i, j, i*j);
```



실습 : 한자 코드 출력하기

▶ 모든 한자 코드 출력하기

- ▶ KSC5601에서 한자 코드의 범위
 - ▶ 2byte 중에서 b1: 0xca-0xfd b2: 0xa1-0xfe

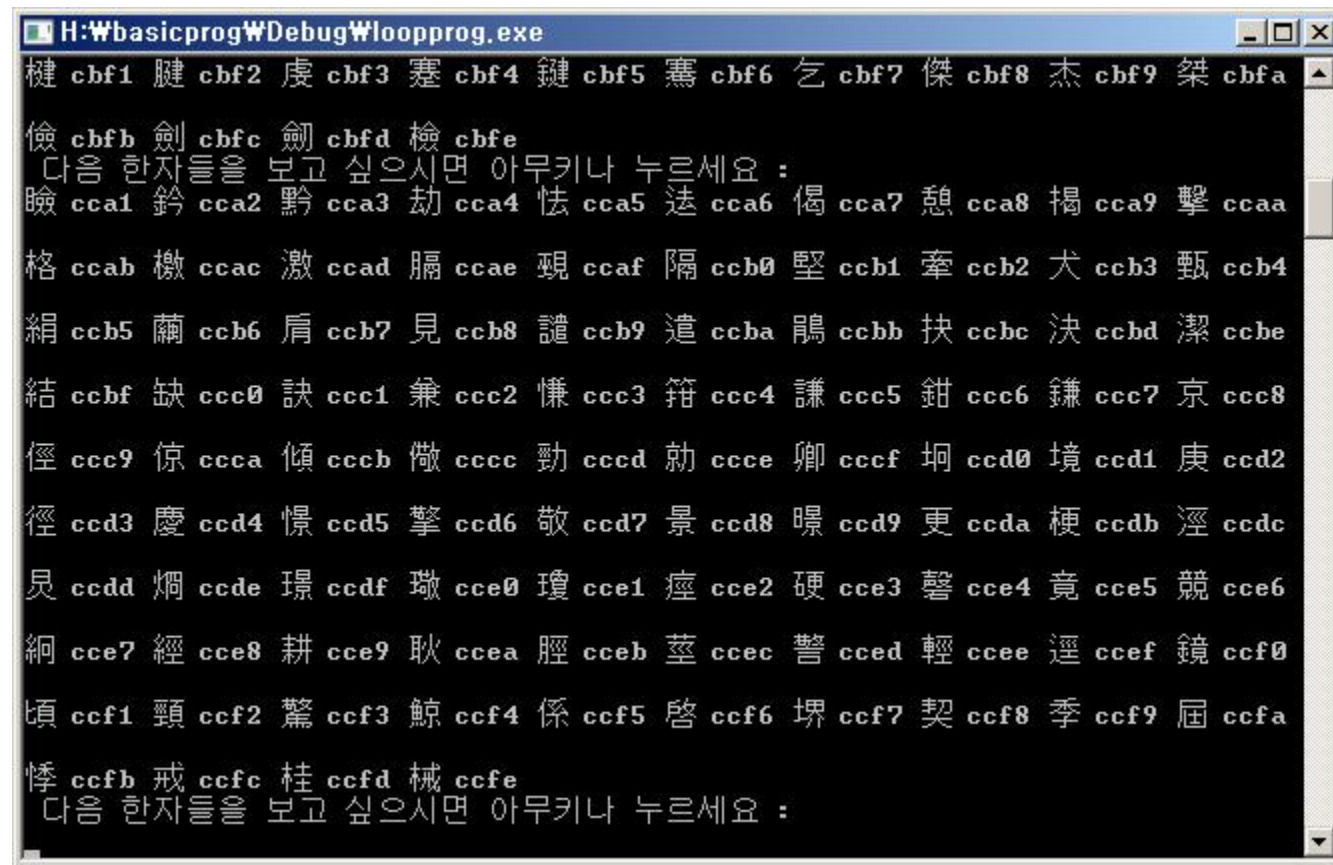
```
#include <stdio.h>

int main(void)
{
    // 한자의 범위 b1 : 0xca-0xfd    b2: 0xa1-0xfe
    unsigned char b1 = 0x00;
    unsigned char b2 = 0x00;

    for( b1=0xca; b1<=0xfd; b1++)
    {
        for( b2 = 0xa1; b2<= 0xfe; b2++ )
        {
            printf("%c%c %2x%2x ", b1, b2, b1, b2);
            if (b2%10 == 0)
                printf("\n");
        }
        printf("\n 다음 한자들을 보고 싶으시면 아무키나 누르세요 : \n");
        _getch();
    }
}
```

실습 : 한자 코드 출력하기

▶ 모든 한자 코드 출력하기



```
H:\Wbasicprog\WDebug\Wloopprog.exe
槌 cbf1 隄 cbf2 虔 cbf3 蹇 cbf4 鍵 cbf5 騫 cbf6 乞 cbf7 傑 cbf8 杰 cbf9 桀 cbfa
儉 cbfb 劍 cbfc 劒 cbfd 檢 cbfe
다음 한자들을 보고 싶으시면 아무키나 누르세요 :
睞 cca1 鈐 cca2 黔 cca3 劫 cca4 怯 cca5 迭 cca6 偈 cca7 憇 cca8 揭 cca9 擊 ccaa
格 ccab 檄 ccac 激 ccad 膈 ccae 覲 ccac 隔 ccb0 堅 ccb1 牽 ccb2 犬 ccb3 甄 ccb4
絹 ccb5 繭 ccb6 肩 ccb7 見 ccb8 譴 ccb9 遣 ccba 鵠 cccb 扶 ccbe 決 ccbe 潔 ccbe
結 ccbf 缺 ccc0 訣 ccc1 兼 ccc2 慊 ccc3 箝 ccc4 謙 ccc5 鉗 ccc6 鎌 ccc7 京 ccc8
徑 ccc9 倥 ccca 傾 ccch 微 cccc 勁 cccd 勛 ccce 卿 cccf 垠 ccd0 境 ccd1 庚 ccd2
徑 ccd3 慶 ccd4 憬 ccd5 擎 ccd6 敬 ccd7 景 ccd8 曠 ccd9 更 ccda 梗 ccdb 涇 ccde
炅 ccdd 烱 ccde 璟 ccdf 璫 cce0 瓊 cce1 瘡 cce2 硬 cce3 磬 cce4 竟 cce5 競 cce6
絳 cce7 經 cce8 耕 cce9 耿 ccea 脛 cceb 莖 ccec 警 cced 輕 ccee 逕 ccef 鏡 ccf0
頃 ccf1 頸 ccf2 驚 ccf3 鯨 ccf4 係 ccf5 啓 ccf6 堺 ccf7 契 ccf8 季 ccf9 屆 ccfa
悽 ccfb 戒 ccfc 桂 ccfd 械 ccfe
다음 한자들을 보고 싶으시면 아무키나 누르세요 :
```

break 문

- ▶ break 문은 반복 루프를 빠져 나오는데 사용된다.

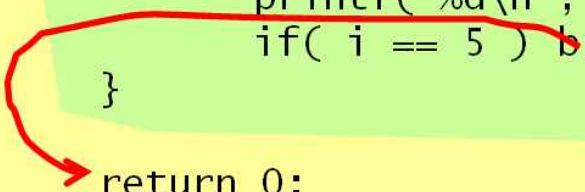
```
#include <stdio.h>

int main(void)
{
    int i;

    for(i=1; i<100; i++)
    {
        printf("%d\n", i);
        if( i == 5 ) break;
    }

    return 0;
}
```

for 반복 루프



예제



// break를 이용하여 무한루프를 탈출한다.

```
#include <stdio.h>
#include <math.h>
```



```
int main(void)
{
```

```
    double v;
```

```
    while(1)
    {
```

```
        printf("실수값을 입력하시오: ");
```

```
        scanf("%lf", &v);
```

```
        if( v < 0.0 )
```

```
            break;
```

```
        printf("%f의 제곱근은 %f입니다.\n", v, sqrt(v));
```

```
    }
```

```
    return 0;
```

```
}
```

실수값을 입력하시오: 9.0
9.000000의 제곱근은 3.000000입니다.
실수값을 입력하시오: 12.0
12.000000의 제곱근은 3.464102입니다.
실수값을 입력하시오: 25.0
25.000000의 제곱근은 5.000000입니다.
실수값을 입력하시오: -1



예제



// break를 이용하여 무한루프를 탈출한다.

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    float grade, sum = 0.0, average;
```

```
    int count = 0;
```

```
    while(1)
```

```
    {
```

```
        printf("학생 성적을 입력하시오: ");
```

```
        scanf("%f", &grade);
```

```
        if( grade < 0.0 )
```

```
            break;
```

```
        count++;
```

```
        sum += grade;
```

```
    }
```

```
    average = sum / count;
```

```
    printf("학생들의 성적의 평균은 %f입니다.\n", average);
```

```
    return 0;
```

```
}
```



학생 성적을 입력하시오: 90

학생 성적을 입력하시오: 90

학생 성적을 입력하시오: 80

학생 성적을 입력하시오: 70

학생 성적을 입력하시오: -1

학생들의 성적의 평균은 82.500000입니다.

goto문의 사용



```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int x, y;
```

```
    for(y = 1; y < 10000; y++)
```

```
    {
```

```
        for(x = 1; x < 50; x++)
```

```
        {
```

```
            if( _kbhit() ) goto OUT;
```

```
            printf("*");
```

```
        }
```

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

```
    }
```

```
OUT:
```

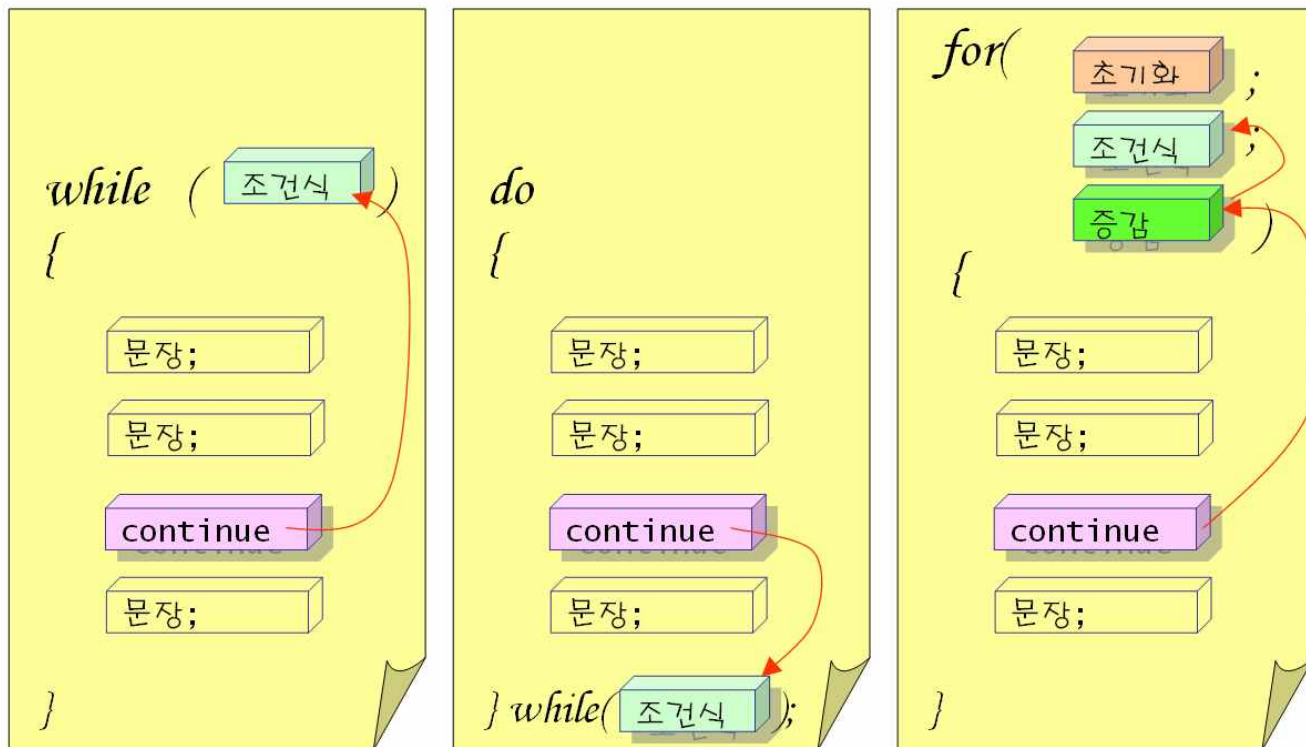
```
    return 0;
```

```
}
```



continue 문

- ▶ 현재의 반복을 중단하고 다음 반복을 시작하게 한다.



예제



```
#include <stdio.h>

int main(void)
{
    int i = 0;
    int sum = 0;

    for(i = 0; i < 100; i++)
    {
        if(i % 2 == 1)
            continue;
        sum += i;
    }
    printf("sum = %d\n", sum);

    return 0;
}
```



sum = 245



예제



// 소문자를 대문자로 변경한다.

#include <stdio.h>

```
int main(void)
{
```

```
    char letter;
```

```
    while(1)
```

```
    {
```

```
        printf("소문자를 입력하시오: ");
```

```
        scanf(" %c", &letter);
```

```
        if( letter == 'Q' )
```

```
            break ;
```

```
        if( letter < 'a' || letter > 'z' )
```

```
            continue ;
```

```
        letter -= 32;
```

```
        printf("변환된 대문자는 %c입니다.\n", letter);
```

```
    }
```

```
    return 0;
```

```
}
```



소문자를 입력하시오: a
변환된 대문자는 A입니다.
소문자를 입력하시오: b
변환된 대문자는 B입니다.
소문자를 입력하시오: c
변환된 대문자는 C입니다.
소문자를 입력하시오: Q

예제



```
// 복리이자계산
#include <stdio.h>

#define RATE 0.07           // 이율
#define INVESTMENT 10000000 // 초기 투자금
#define YEARS 10           // 투자 기간

int main(void)
{
    int i;
    double total = INVESTMENT; // 원리금 합계

    printf("=====\n");
    printf("연도 원리금\n");
    printf("=====\n");

    for(i = 1; i <= YEARS; i++)
    {
        total = total * ( 1 + RATE ); // 새로운 원리금 계산
        printf("%2d %10.1f\n", i, total);
    }

    return 0;
}
```



```
=====  
연도 원리금  
=====  
1 10700000.0  
2 11449000.0  
3 12250430.0  
4 13107960.1  
5 14025517.3  
6 15007303.5  
7 16057814.8  
8 17181861.8  
9 18384592.1  
10 19671513.6
```

예제



```
#include <stdio.h>
#define START_DAY      3    // 첫번째 날이 수요일
#define DAYS_OF_MONTH  31   // 달의 일수

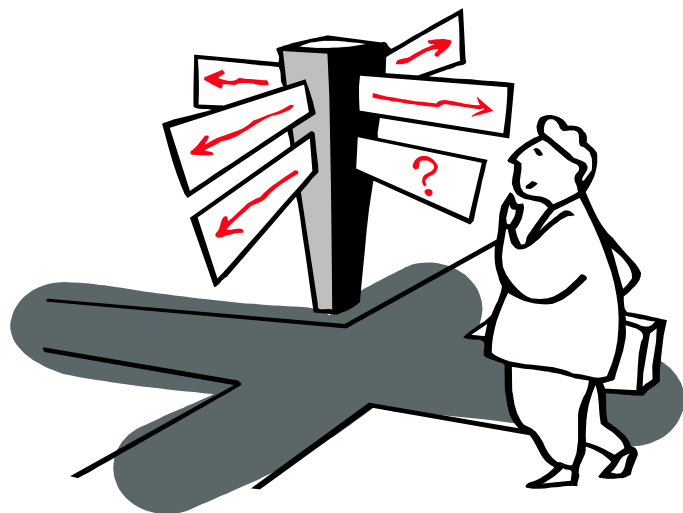
int main(void)
{
    int day, date;
    printf("=====\n");
    printf("일 월 화 수 목 금 토\n");
    printf("=====\n");

    for(day = 0; day < START_DAY ; day++)    // 월요일부터 수요일까지
        printf(" ");                        // 공백 출력
    for(date = 1; date <= DAYS_OF_MONTH ; date++)
    {
        if( day == 7 )
        {
            day = 0;    // 일요일이면 줄바꿈을 출력
            printf("\n");
        }
        day++;
        printf("%2d ", date);                // 날을 출력한다.
    }
    printf("\n=====\n");
    return 0;
}
```



```
=====
일 월 화 수 목 금 토
=====
           1  2  3  4
5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
=====
```





Chapter 07이 끝났습니다. 질문 있으신지요?