

프로그래밍언어

### 3. 프로그램 실행 제어, 반복문, 다차원 배열과 행렬



교수 김 영 탁

영남대학교 정보통신공학과

(Tel : +82-53-810-2497; E-mail : ytkim@yu.ac.kr)

# Outline

- ◆ 제어문, 조건문이란 ?
- ◆ if, if – else, 다중 if 문
- ◆ switch, break, continue
- ◆ goto
- ◆ 반복문이란 ?
- ◆ while, do – while, for
- ◆ 중첩 반복문 (nested loop)
- ◆ break, continue
- ◆ 2차원 배열(two-dimensional Array)과 행렬(Matrix)



**조건식, 조건문**

# 조건식 관련 연산자

## ◆ 조건식 관련 연산자 (conditional operator)

연산자의 분류	연산자	의미, 예
관계연산자 (relationship)	>	$a > b$ : a가 b보다 크면 true, 아니면 false
	>=	$a \geq b$ : a가 b보다 같거나 크면 true, 아니면 false
	<	$a < b$ : a가 b보다 작으면 true, 아니면 false
	<=	$a \leq b$ : a가 b보다 같거나 작으면 true, 아니면 false
	==	$a == b$ : a와 b가 같으면 true, 아니면 false
	!=	$a != b$ : a와 b가 다르면 true, 아니면 false
논리연산자 (logical)	&& (논리 곱)	$A \&\& B$ : A와 B가 모두 true이면 true, 아니면 false
	(논리 합)	$A    B$ : A나 B 둘 중 하나가 true이면 true, 아니면 (즉, A와 B 모두 false이면) false
	!	$!A$ : A가 true이면 false, A가 false이면 true
조건연산자 (conditional)	?	조건에 따라 선택 $\text{max} = (x \geq y) ? x : y$ ; 만약 x가 y보다 같거나 크면 x를 선택, 아니면 y를 선택



# 논리 연산자를 사용한 조건식 표현

주어진 조건	산술 연산자를 사용한 수식의 표현
성적 (score)이 90보다 같거나 높고, 95보다 낮은 경우	<pre>if (score &gt;= 90) &amp;&amp; (score &lt; 95))     printf("Your score is A");</pre>
윤년 (leap year)의 조건: 년도가 4의 배수이며 100의 배수가 아니거나, 또는 400의 배수이면 윤년	<pre>if ((year % 4 == 0) &amp;&amp; (year % 100 != 0))        (year % 400 == 0))     printf("%d year is leap year \n", year);</pre>
기온이 30도 이상이며, 날씨가 화창할 때	<pre>if ((temp &gt;= 30) &amp;&amp; (weather == "SUNNY"))     printf("It's good for picnic !!\n");</pre>



# if문 예제

// if 문을 사용하여 절대값을 구하는 프로그램

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

```
int main(void)
```

```
{
```

```
    int number;
```

→ 

```
    printf("정수를 입력하시오:");  
    scanf("%d", &number);
```

만약  
사용자가 -5를  
입력하였다면

```
    if( number < 0 )  
        number = -number;
```

-5 < 0이므로  
해당 조건문  
실행

```
    printf("절대값은 %d 입니다.\n", number);
```

```
    return 0;
```

```
}
```

정수를 입력하시오.  
-5  
절대값은 5 입니다.



# if-else 문 예제

```
if ( score >= 60 )
```

```
    printf("합격입니다.\n");
```

```
else
```

```
    printf("불합격입니다.\n");
```

score가 60이상이면 실행

score가 60미만이면 실행

```
if ( score >= 60 )
```

```
{
```

```
    printf("합격입니다.\n");
```

```
    printf("장학금도 받을 수 있습니다.\n");
```

```
}
```

```
else
```

```
{
```

```
    printf("불합격입니다.\n");
```

```
    printf( " 더욱 열심히 공부하세요.\n");
```

```
}
```

score가 60이상이면 실행

score가 60미만이면 실행



# 복잡한 조건식의 예제

```
// 윤년 판단 프로그램  
#include <stdio.h>
```

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

```
    int year;
```

```
    printf("연도를 입력하시오: ");  
    scanf("%d", &year);
```


```
    if((year % 4 == 0 && year % 100 != 0) || year % 400 == 0)  
        printf("%d년은 윤년입니다.\n", year);
```

```
    else
```

```
        printf("%d년은 윤년이 아닙니다.\n", year);
```

```
    return 0;
```

```
}
```



연도를 입력하시오: 2005  
2005년은 윤년이 아닙니다.

PASS



# 중첩 if

(a) if 문 내부에 다른 if 문이 포함된 경우

```
if( score >= 80 )
{
    if( score >= 90 ) // 학점이 80 보다 같거나 크고, 또한 90 보다 같거나 큰 경우
        printf("당신의 학점은 A입니다.\n");
}
```

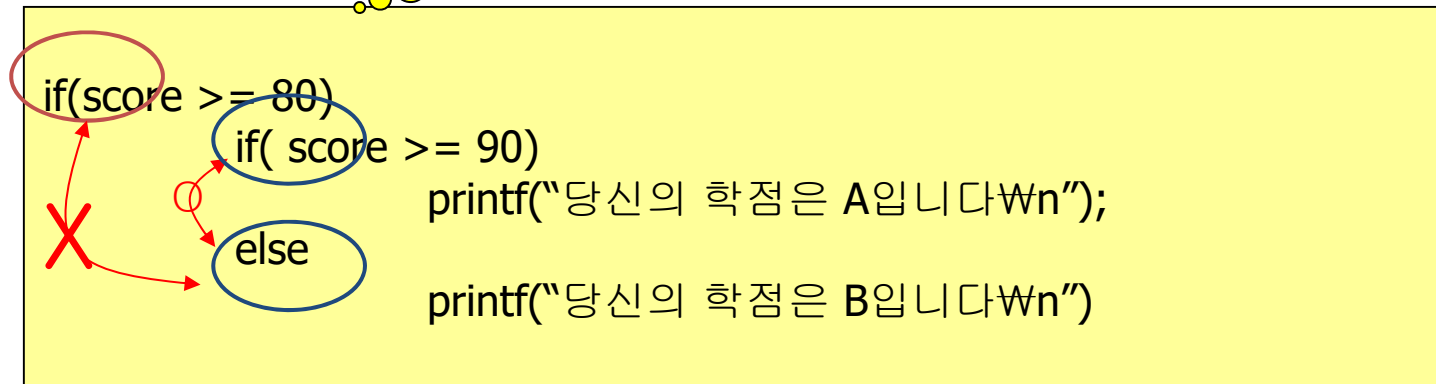
(b) if 문 내부에 if – else 문이 포함된 경우

```
if( score >= 80 )
{
    if( score >= 90 ) // 학점이 80보다 같거나 크고, 또한 90보다 같거나 큰 경우
        printf("당신의 학점은 A입니다.\n");
    else // 학점이 80보다 같거나 크지만 90보다 작은 경우
        printf("당신의 학점은 B입니다.\n");
}
```



# if와 else의 매칭 문제

else 절은 가장 가까운  
if절과 매치된다.



```
if( score >= 80 )
{
    if( score >= 90 )
        printf("당신의 학점은 A입니다.\n");
}
else
{
    printf("당신의 학점은 A나 B가 아닙니다.\n");
}
```

만약 다른 if절과 else 절  
을 매치 시키려면 중괄  
호를 사용하여 블록으로  
묶는다.



# 정수 3개중의 최소값 찾기

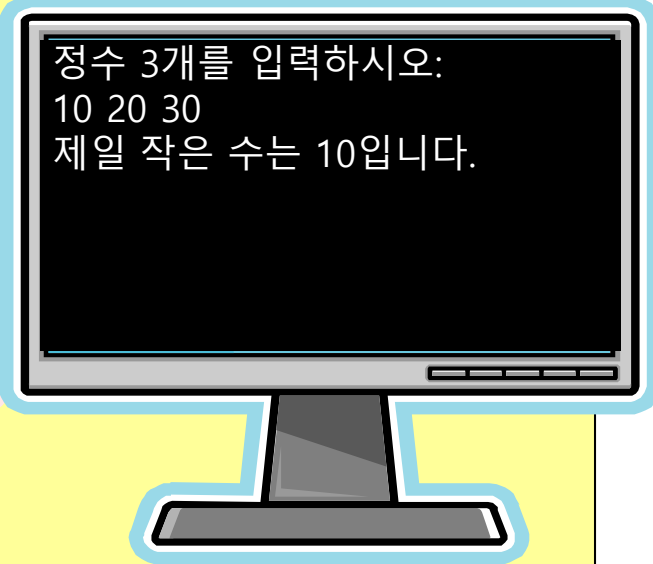
```
#include <stdio.h>
int main(void)
{
    int n1, n2, n3, min;

    printf("정수 3개를 입력하시오:\n");
    scanf("%d %d %d", &n1, &n2, &n3);

    if( n1 < n2 )
        if( n1 < n3 )
            min = n1;
        else
            min = n3;
    else
        if( n2 < n3 )
            min = n2;
        else
            min = n3;

    printf("제일 작은 수는 %d입니다\n", min);
    return 0;
}
```

10 < 20 이고 10  
< 30 이므로 실행



정수 3개를 입력하시오:  
10 20 30  
제일 작은 수는 10입니다.



# 문자 분류 예제

```
// 문자들을 분류하는 프로그램
#include <stdio.h>

int main(void)
{
    char ch;

    printf("문자를 입력하시오: ");
    scanf("%c", &ch);

    if( ch >= 'A' && ch <= 'Z' )
        printf("%c는 대문자입니다.\n", ch);
    else if( ch >= 'a' && ch <= 'z' )
        printf("%c는 소문자입니다.\n", ch);
    else if( ch >= '0' && ch <= '9' )
        printf("%c는 숫자입니다.\n", ch);
    else
        printf("%c는 기타문자입니다.\n", ch);

    return 0;
}
```

조건 만족

문자를 입력하시오: c  
c는 소문자입니다.



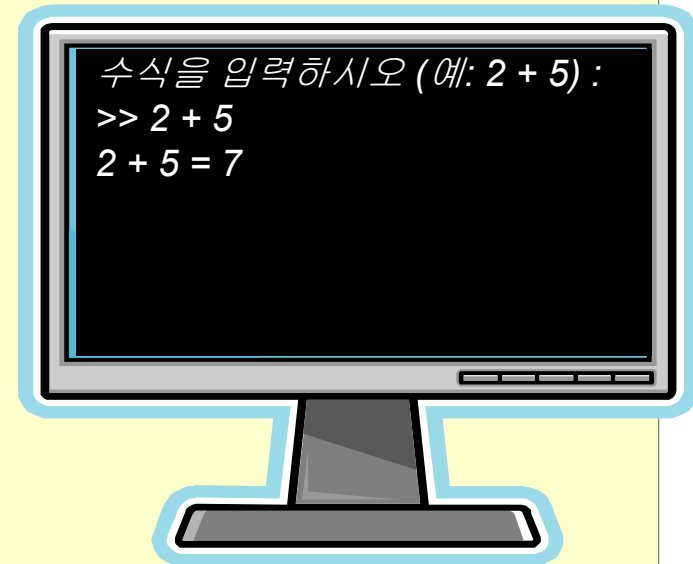
# 간단한 수식 계산

```
#include <stdio.h>
int main(void)
{
    char op;
    int x, y, result;

    printf("수식을 입력하십시오");
    printf("(예: 2 + 5) : \n");
    printf(">> ");
    scanf("%d %c %d", &x, &op, &y);

    if( op == '+' )
        result = x + y;
    else if( op == '-' )
        result = x - y;
    else if( op == '*' )
        result = x * y;
    else if( op == '/' )
        result = x / y;
    else if( op == '%' )
        result = x % y;
    else
        printf("지원되지 않는 연산자입니다. ");

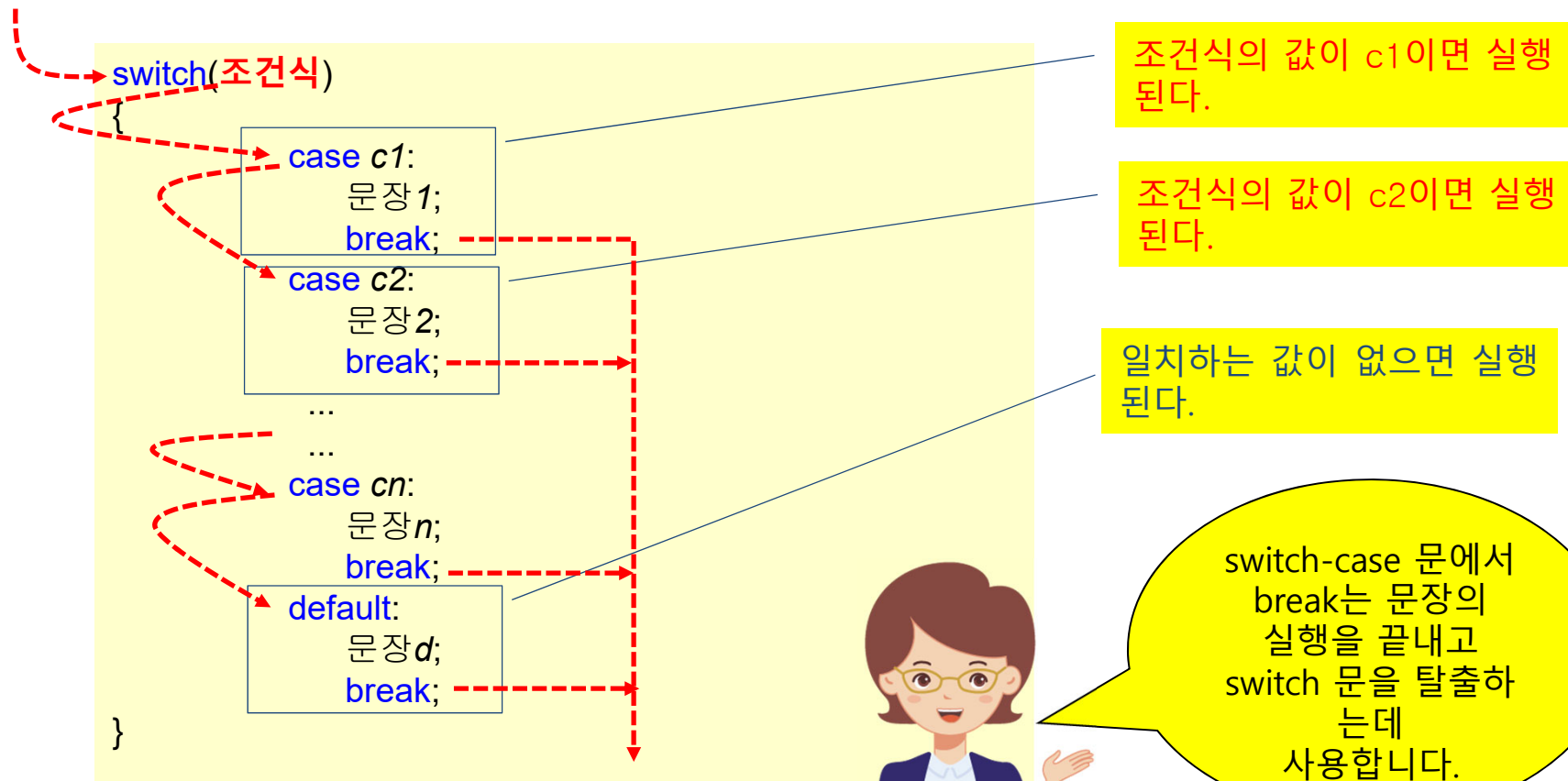
    printf("%d %c %d = %d ", x, op, y, result);
    return 0;
}
```



**switch – case 구조**

# switch 문

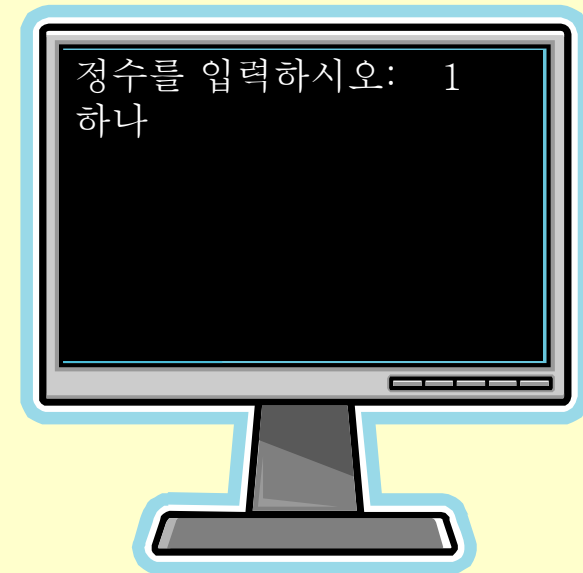
## ◆ 여러가지 경우 중에서 하나를 선택하는데 사용



# 예제

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

    printf("정수를 입력하시오:");
    scanf("%d", &number);
    switch(number)
    {
        case 0:
            printf("없음\n");
            break ;
        case 1:
            printf("하나\n");
            break ;
        case 2:
            printf("둘\n");
            break ;
        default:
            printf("많음\n");
            break;
    }
}
```

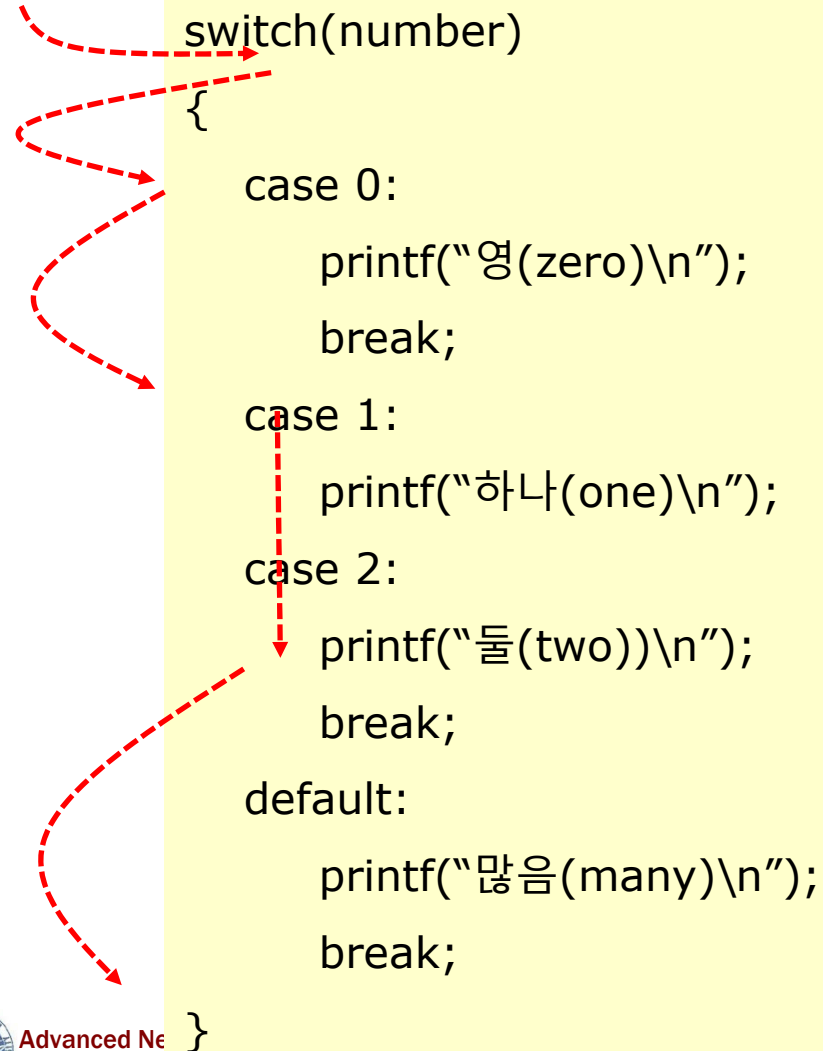




# break가 생략되는 경우

number = 1

```
switch(number)
{
    case 0:
        printf("영(zero)\n");
        break;
    case 1:
        printf("하나(one)\n");
    case 2:
        printf("둘(two)\n");
        break;
    default:
        printf("많음(many)\n");
        break;
}
```



조건에 맞는  
case로 부터  
break를 만날 때  
까지 계속 문장  
을 실행합니다.



# 의도적인 break생략

```
switch(number)
{
    case 0:
        printf("없음\n");
        break;
    case 1:
        printf("하나\n");
        break;
    case 2:
    case 3:
    case 4:
        printf("두 서너개\n");
        break;
    default:
        printf("많음\n");
        break;
}
```

2개의 경우를 하나로 묶어서 처리하기 위하여 이러한 기법을 사용



# default 문

◆ 어떤 case문과도 일치되지 않는 경우에 default를 선택

number = 5

```
switch(number)
{
    case 0:
        printf("영(zero)\n");
        break;
    case 1:
        printf("하나(one)\n");
        break;
    case 2:
        printf("둘(two)\n");
        break;
    default:
        printf("많음(many)\n");
        break;
}
```

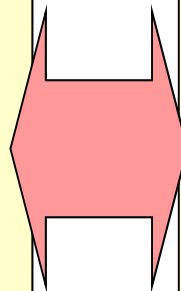
switch – case 문에  
서 조건이 맞는  
case문이 아무것도  
없을 때, default 부  
분이 실행됩니다.



# switch 문과 if-else 문

```
int main(void)
{
    int number;
    scanf("%d", &number);

    if( number == 0 )
        printf("없음\n");
    else if( number == 1 )
        printf("하나\n");
    else if( number == 2 )
        printf("둘\n");
    else
        printf("많음\n");
}
```



```
switch(number)
{
    case 0:
        printf("없음\n");
        break;
    case 1:
        printf("하나\n");
        break;
    case 2:
        printf("둘\n");
        break;
    default:
        printf("많음\n");
        break;
}
```

# switch 문에서 주의할 점

```
switch(number)
{
    case x: // 변수는 사용할 수 없다.
        printf("x와 일치합니다. ");
        break;
    case (x+2): // 변수가 들어간 수식은 사용할 수 없다.
        printf("수식과 일치합니다. ");
        break;
    case 0.001: // 실수는 사용할 수 없다.
        printf("실수");
        break;
    case "001": // 문자열은 사용할 수 없다.
        printf("문자열");
        break;
}
```

switch – case 문의  
case에는 정수형  
상수 (integer  
constant) 만 지정  
될 수 있습니다.



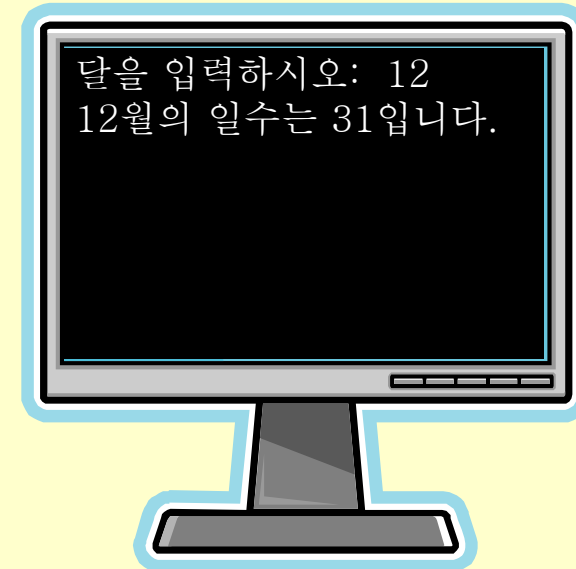
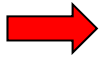
# 예제

```
// 달의 일수를 계산하는 프로그램
#include <stdio.h>

int main(void)
{
    int month, days;

    printf("달을 입력하시오: ");
    scanf("%d", &month);

    switch(month)
    {
        case 2:
            days = 28;
            break;
        case 4:
        case 6:
        case 9:
        case 11:
            days = 30;
            break;
        default:
            days = 31;
            break;
    }
    printf("%d월의 일수는 %d입니다.\n", month, days);
    return 0;
}
```



# 실습: 산술 계산기

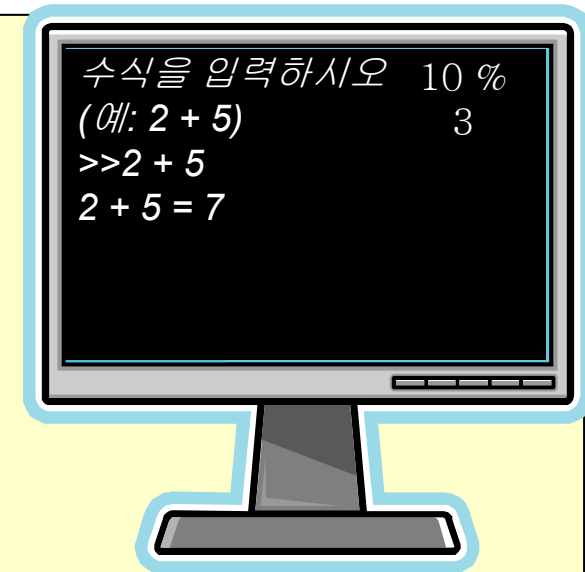
```
#include <stdio.h>

int main(void)
{
    char op;
    int x, y, result;

    printf("수식을 입력하십시오");
    printf("(예: 2 + 5) ");
    printf(">>");
    scanf("%d %c %d", &x, &op, &y);
    switch(op)
    {
        case '+':
            result = x + y;
            break;

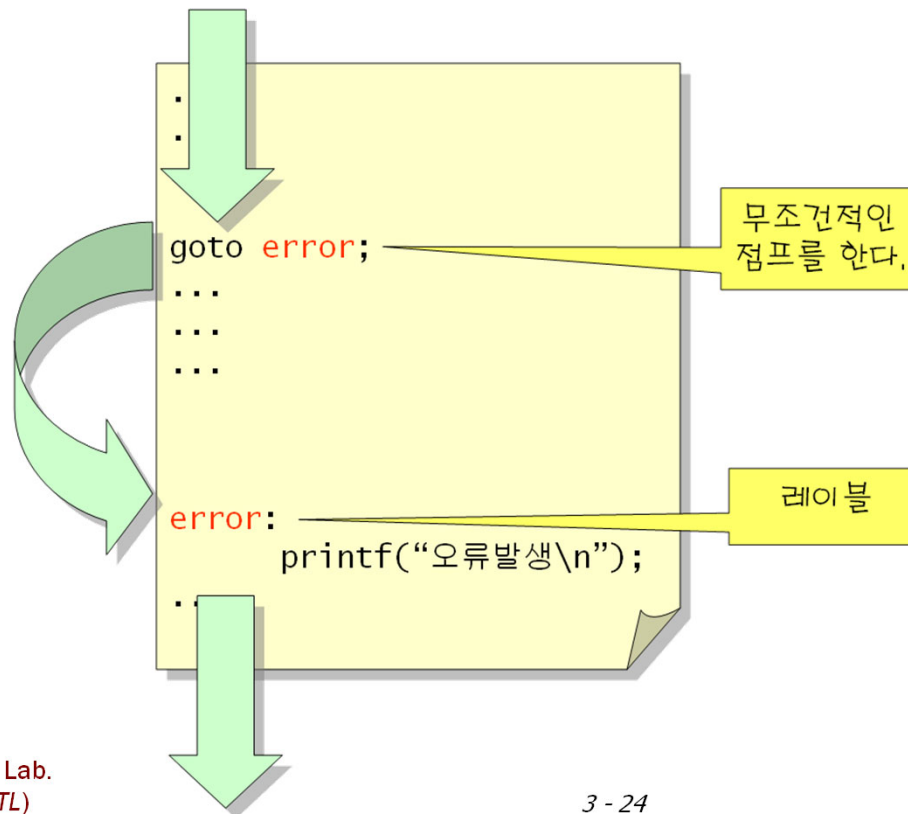
        case '-':
            result = x - y;
            break;

        ...
        default:
            printf("지원되지 않는 연산자입니다. ");
            break;
    }
    printf("%d %c %d = %d ", x, op, y, result);
    return 0;
}
```



# Goto문

- ◆ 조건없이 어떤 위치로 점프
- ◆ 사용하지 않는 것이 좋음 !!  
(운영체제 내부의 프로그래밍 이전에는 사용 금지)



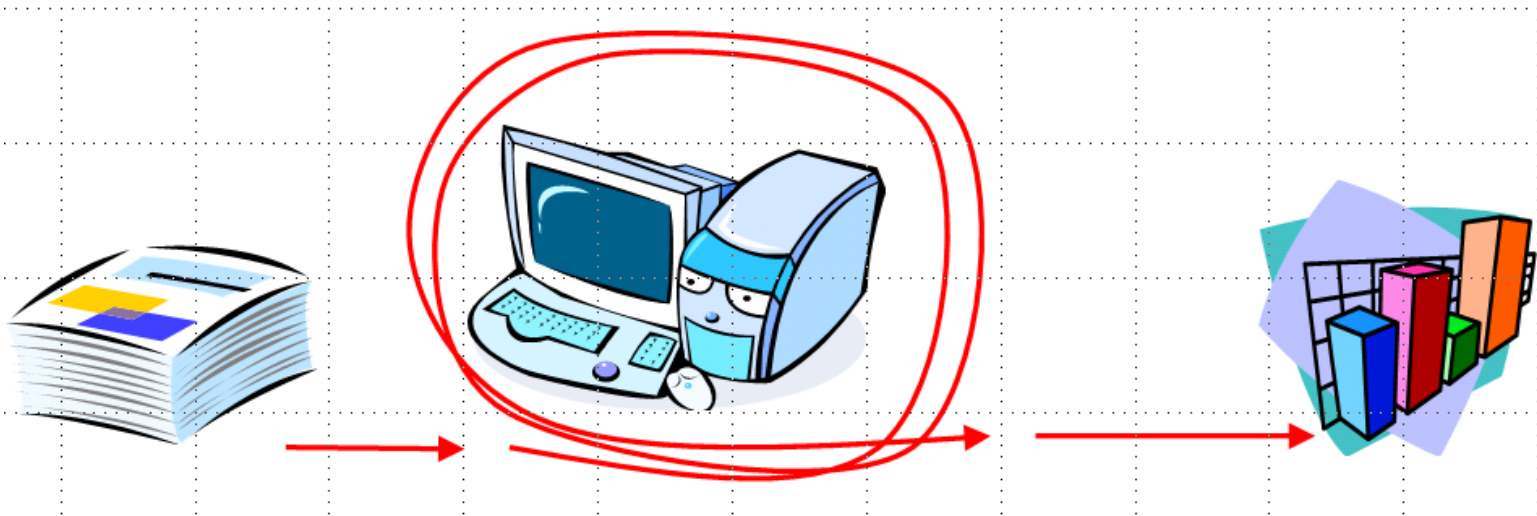


**while** 반복문,  
**do-while** 반복문

# 반복문

Q) 반복 구조는 왜 필요한가?

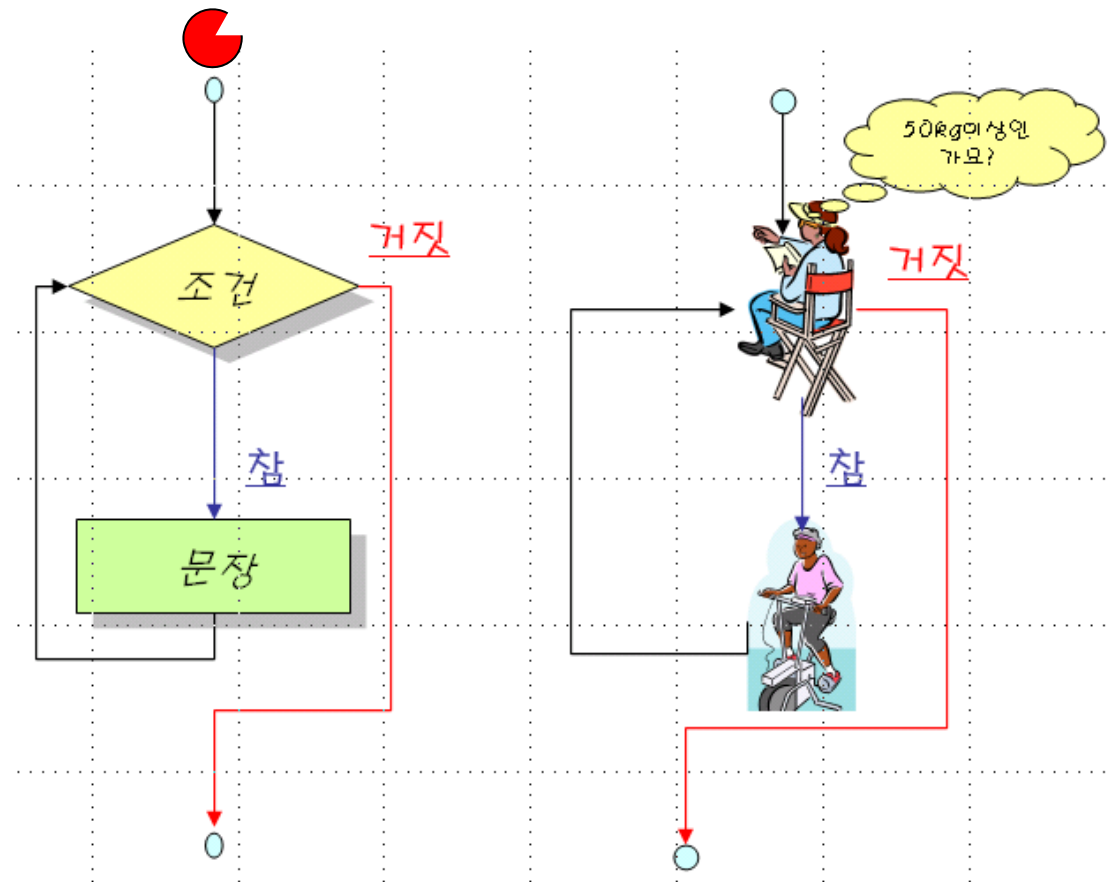
A) 같은 처리 과정을 되풀이하는 것이 필요하기 때문이다.  
학생 30명의 평균 성적을 구하려면 같은 과정을  
30번 반복하여야 한다.



# while 반복문

◆ 주어진 조건이 만족되는 동안 문장들을 반복 실행한다.

```
초기식;  
while( 조건식 )  
{  
    문장;  
    증감식;  
}
```



# 예제

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

```
int main(void)
```

```
{
```

```
    int meter=0;  
    int i = 0;
```

```
    while(i < 3)
```

```
    {
```

```
        meter = i * 1609;
```

```
        printf("%d 마일은 %d 미터입니다\n", i, meter);
```

```
        i++; // i = i+1 과 동일
```

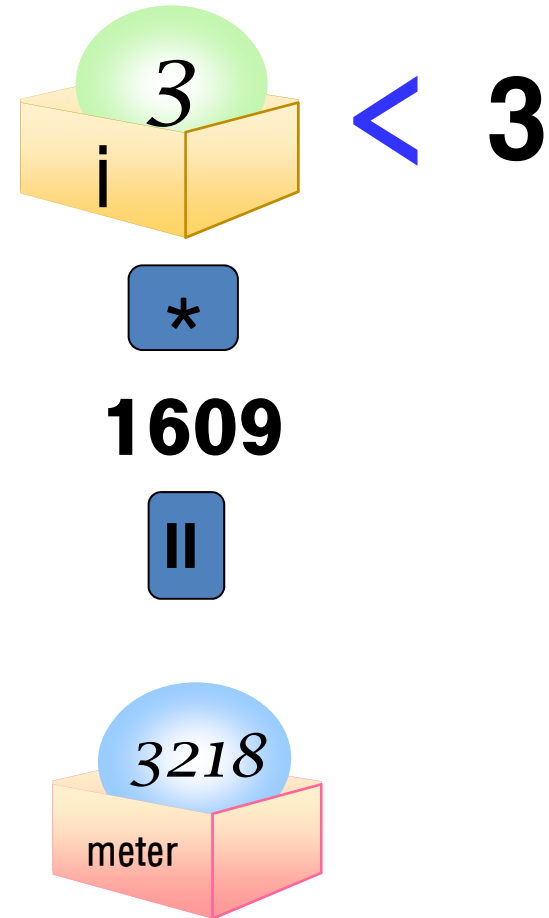
```
    }
```

```
    return 0;
```

```
}
```

i 값이 3으로 증가하였지만 조건에 만족 하지 않아  
출력하지 못하고 프로그램을 종료한다.

0마일은 0미터 입니다.  
1마일은 1609미터 입니다.  
2마일은 3218미터 입니다.



# 예제

```
// while 문을 이용한 구구단 출력 프로그램
```

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

```
int main(void)
```

```
{
```

```
    int n;
```

```
    int i = 1;
```

```
    printf( " 출력하고 싶은 단: ");
```

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

```
    while (i <= 9)
```

```
    {
```

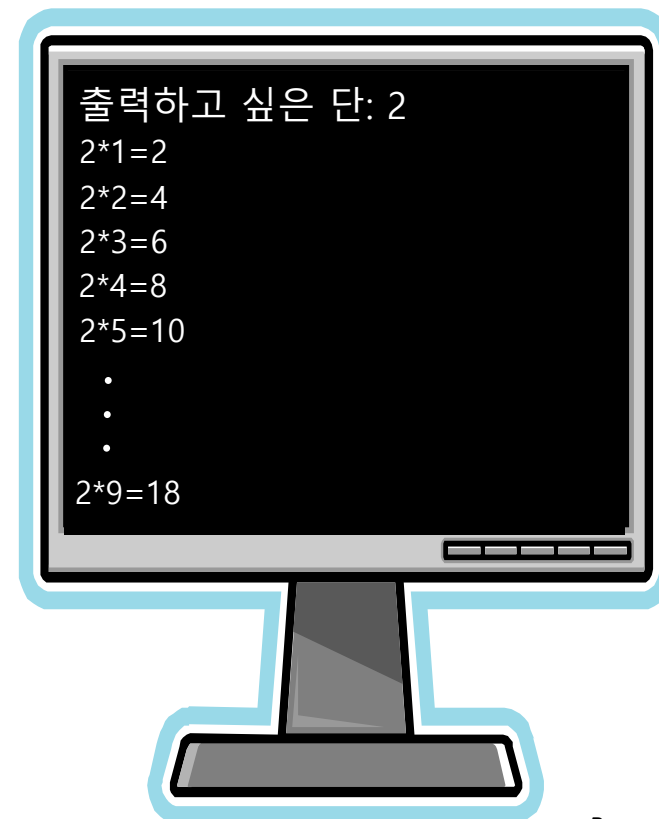
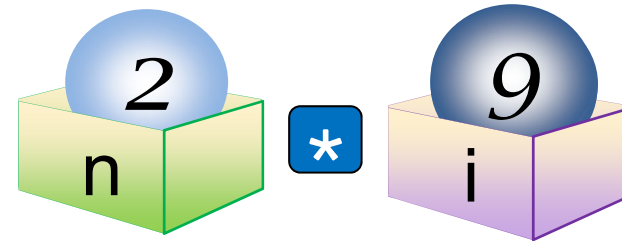
```
        → printf("%d * %d = %d \n", n, i, n * i);
```

```
        i++; // i = i+1 과 동일
```

```
    }
```

```
    return 0;
```

```
}
```



# 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)
```

```
    {
```

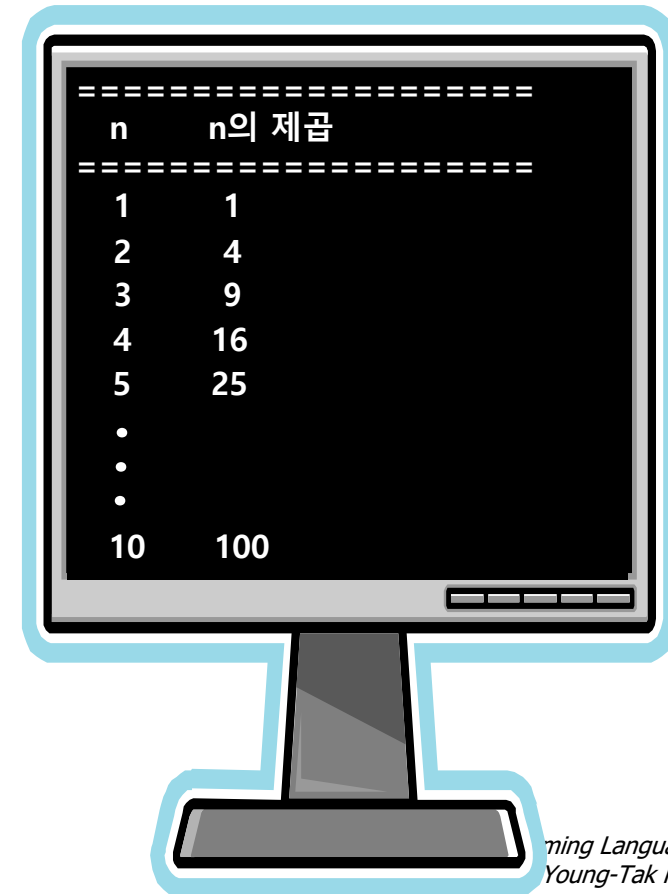
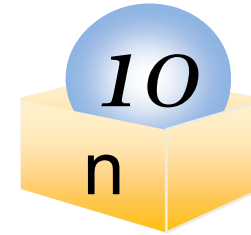
```
        → printf("%5d  %5d \n", n, n*n);
```

```
        n = n+1;
```

```
    }
```

```
    return 0;
```

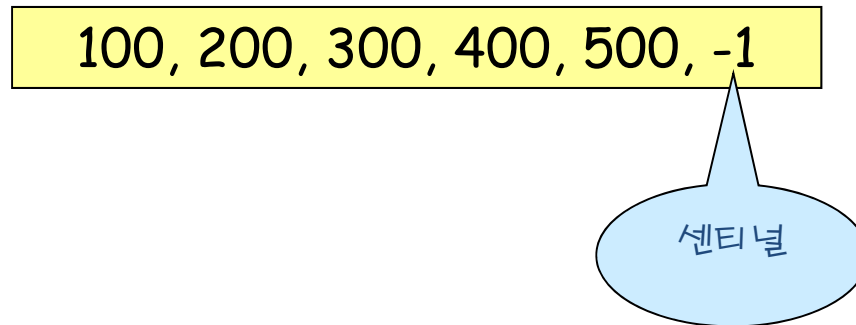
```
}
```



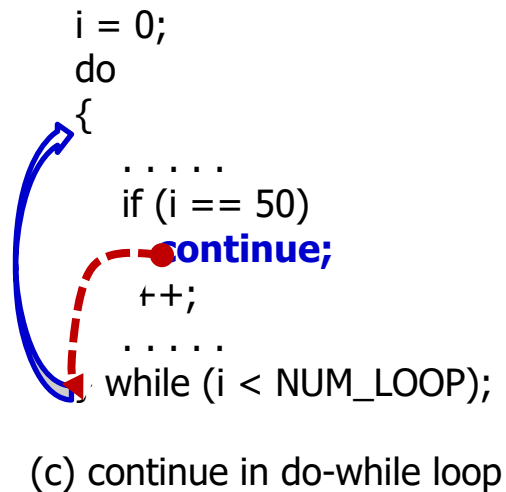
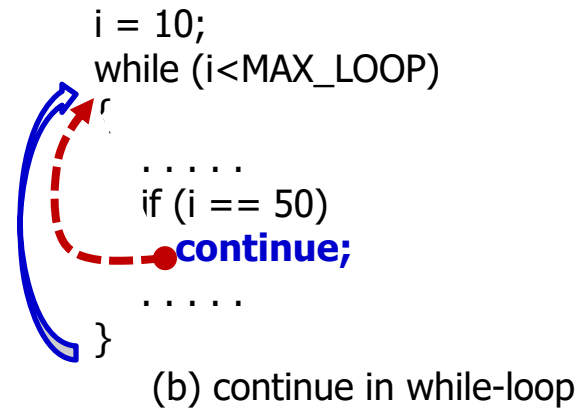
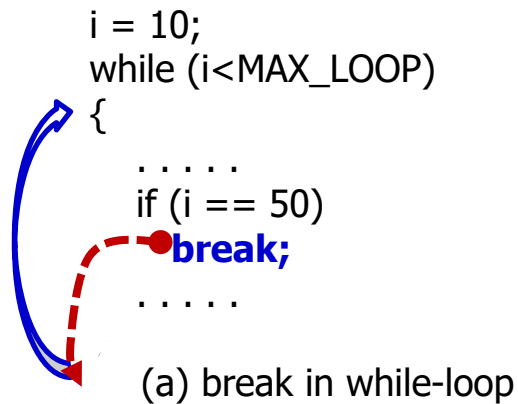
# 센티넬(sentinel, 보초) 의 이용

## ◆ 센티넬

- 입력되는 데이터의 끝을 알리는 특수한 값
- 정상적인 범위를 벗어나는 값으로 설정함



# while 반복문에서의 break와 continue





```

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

void main(void)
{
    int grade = 0, n = 0; // 필요한 변수 선언 및 초기화
    double sum = 0.0, average = 0.0;

    printf( " 학생들의 성적을 입력하십시오 (종료 시 음수 입력)\n");

    // 성적을 입력 받아서 합계를 구하고 학생 수를 센다.
    while (grade >= 0)
    {
        printf("성적을 입력하십시오: ");
        scanf("%d", &grade);
        if (grade < 0)
            break;
        sum += grade;
        n++;
    }
    // 평균을 계산하고 화면에 출력한다.
    average = sum / n;
    printf("성적의 평균은 %lf입니다.\n", average);
}

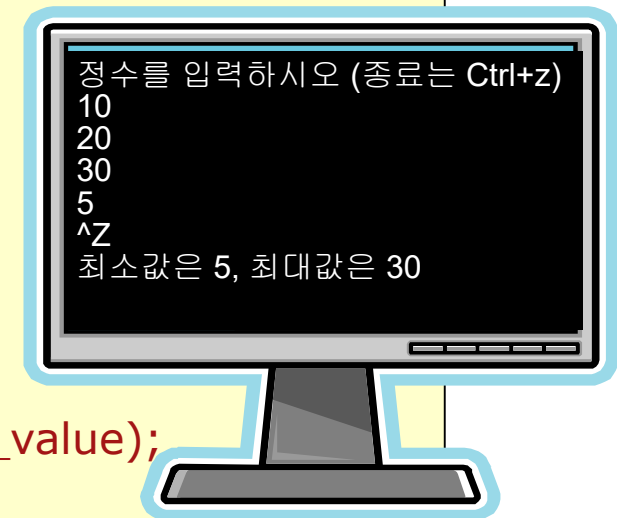
```



# 예제: EOF(Ctrl+z), 최댓값, 최솟값

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

int main(void)
{
    int number, min_value = INT_MAX, max_value = INT_MIN;
    printf("정수를 입력하시오 (종료는 Ctrl+z)\n");
    while(scanf("%d", &number) != EOF)
    {
        if( number < min_value )
            min_value = number;
        if(number > max_value)
            max_value = number;
    }
    printf("최소값은 %d, 최대값은 %d", min_value, max_value);
    return 0;
}
```



# while 반복문에서 주의할 점

```
int i = 1;
while(i < 10)
{
    printf("반복 중입니다\n");
    i--;
}
```

변수가 증가 아니라 감소

```
int i = 0;
while(i < 3)
    printf("반복 중입니다\n");
    i++;
```

반복 루프에 포함되어 있지  
않다. { }로 블록 설정하지  
않았음

```
int i = 0;
while(i < 3) ;
{
    printf("반복 중입니다\n");
    i++;
}
```

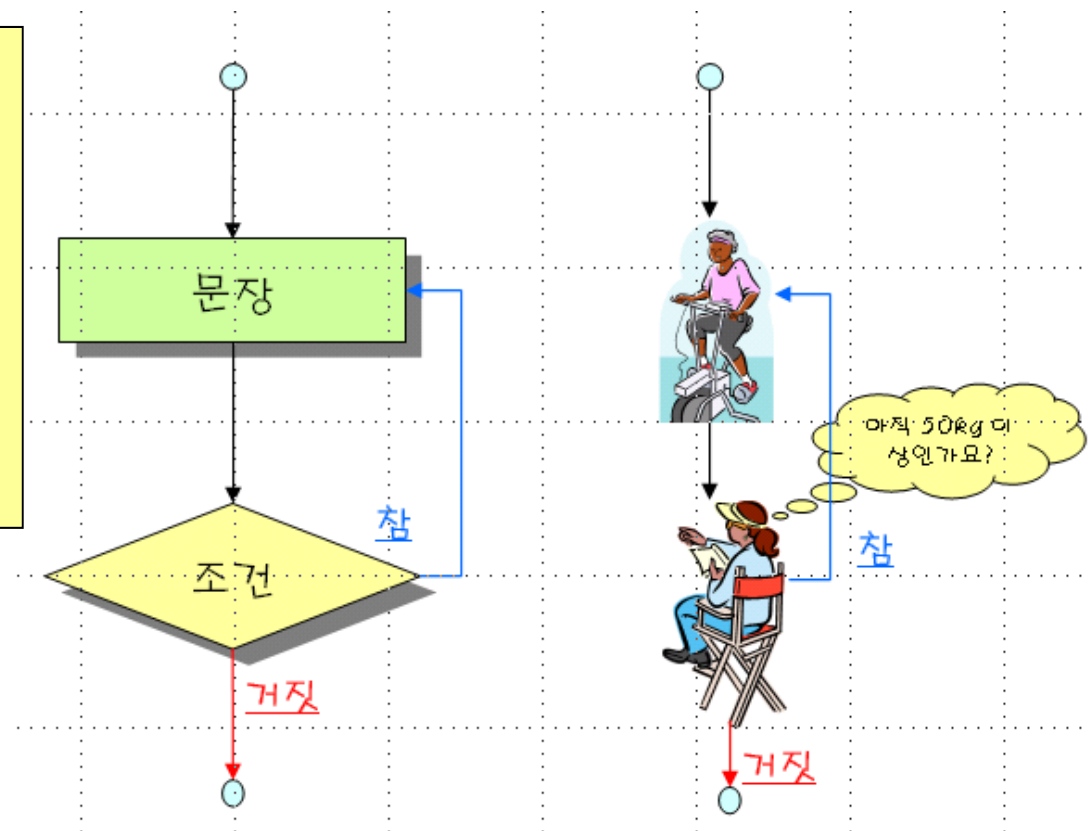
조건뒤에 ;이 있음



# do...while 반복문

## ◆ 반복 조건을 루프의 끝에서 검사

```
초기식;  
do  
{  
    문장;  
    증감식;  
}  
while(조건식);
```



# 예제

```
// do..while 문을 이용한 메뉴  
#include <stdio.h>
```

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

```
    int i = 0;
```

```
    do
```

```
    {
```

```
        printf("1---새로 만들기\n");
```

```
        printf("2---파일 열기\n");
```

```
        printf("3---파일 닫기\n");
```

```
        printf("하나를 선택 하시오:\n");
```

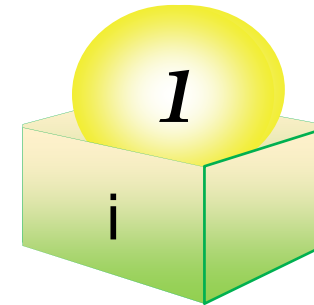
```
        scanf("%d", &i);
```

```
    } while(i < 1 || i > 3);
```

```
    printf("선택된 메뉴=%d\n", i);
```

```
    return 0;
```

```
}
```



```
1---새로 만들기  
2---파일 열기  
3---파일 닫기  
하나를 선택 하시오  
선택된 메뉴=1
```

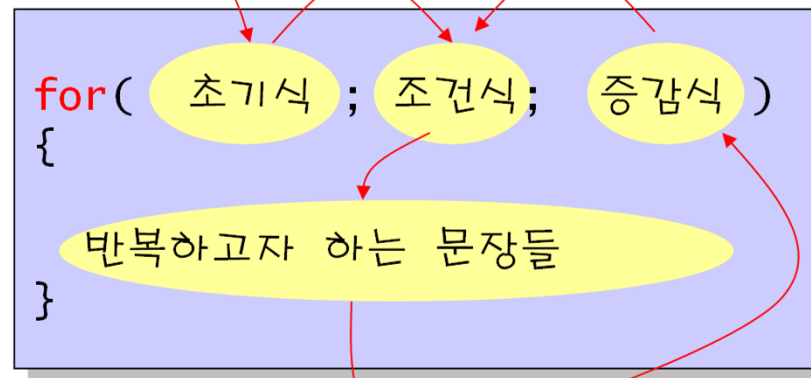


**for 반복문**

# for 반복문

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


- ① 초기식을 실행한다.
- ② 반복 조건을 나타내는 조건식을 계산한다.
- ③ 수식의 값이 거짓이면 **for** 문의 실행이 종료된다.
- ④ 수식의 값이 참이면 문장이 실행된다.
- ⑤ 증감식을 실행하고 ②로 돌아간다.



# for문의 실행과정


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

1번째 루프  
i값은



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

2-10번째 루프  
i값은



...

...

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

11번째 루프  
i값은





# 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
```

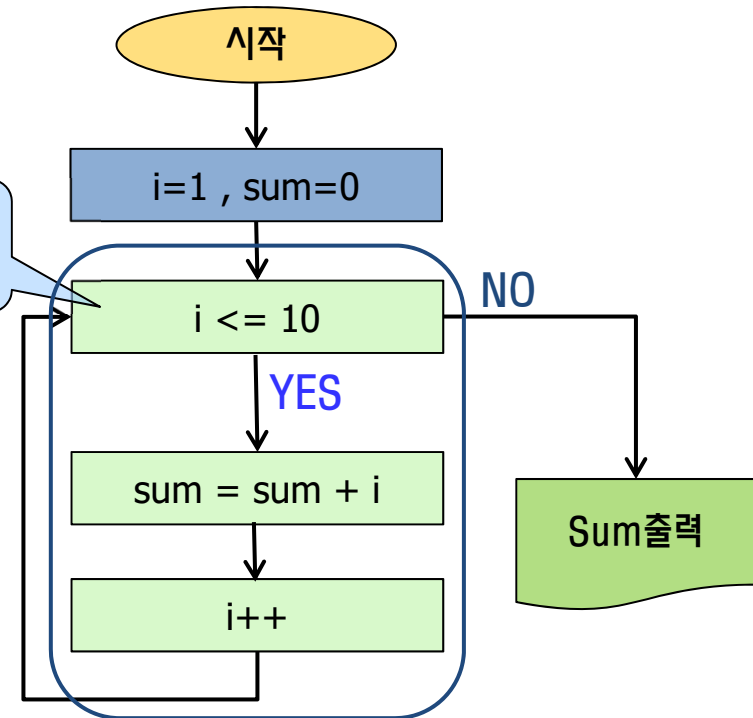
```
    }
```

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

```
    return 0;
```

```
}
```

i가 10보다 작거나 같은 때  
까지 10번 반복



1부터 10까지 정수의 합 = 55



# 예제

```
// 반복을 이용한 세제곱값구하기
```

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

```
int main(void)
```

```
{
```

```
    int i, n;
```

```
    printf("정수를 입력하시오:");
```

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

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

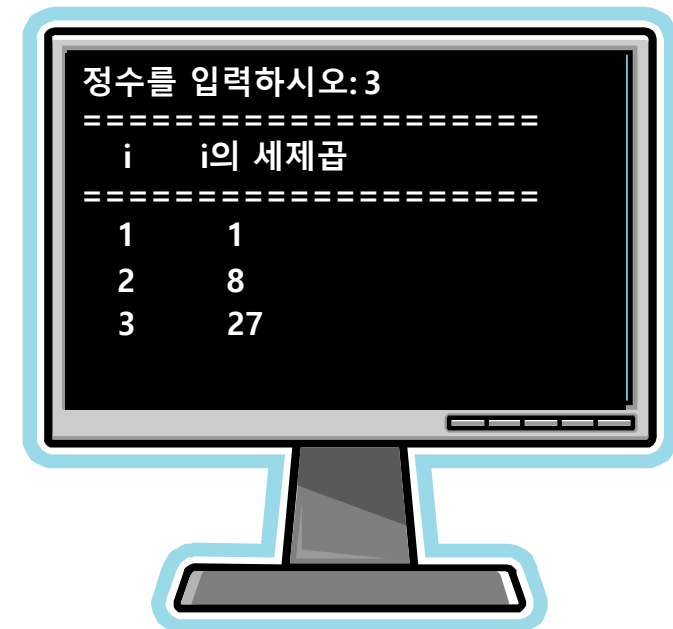
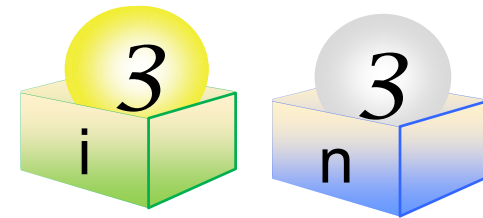
```
    printf("   i       i의 세제곱\n");
```

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

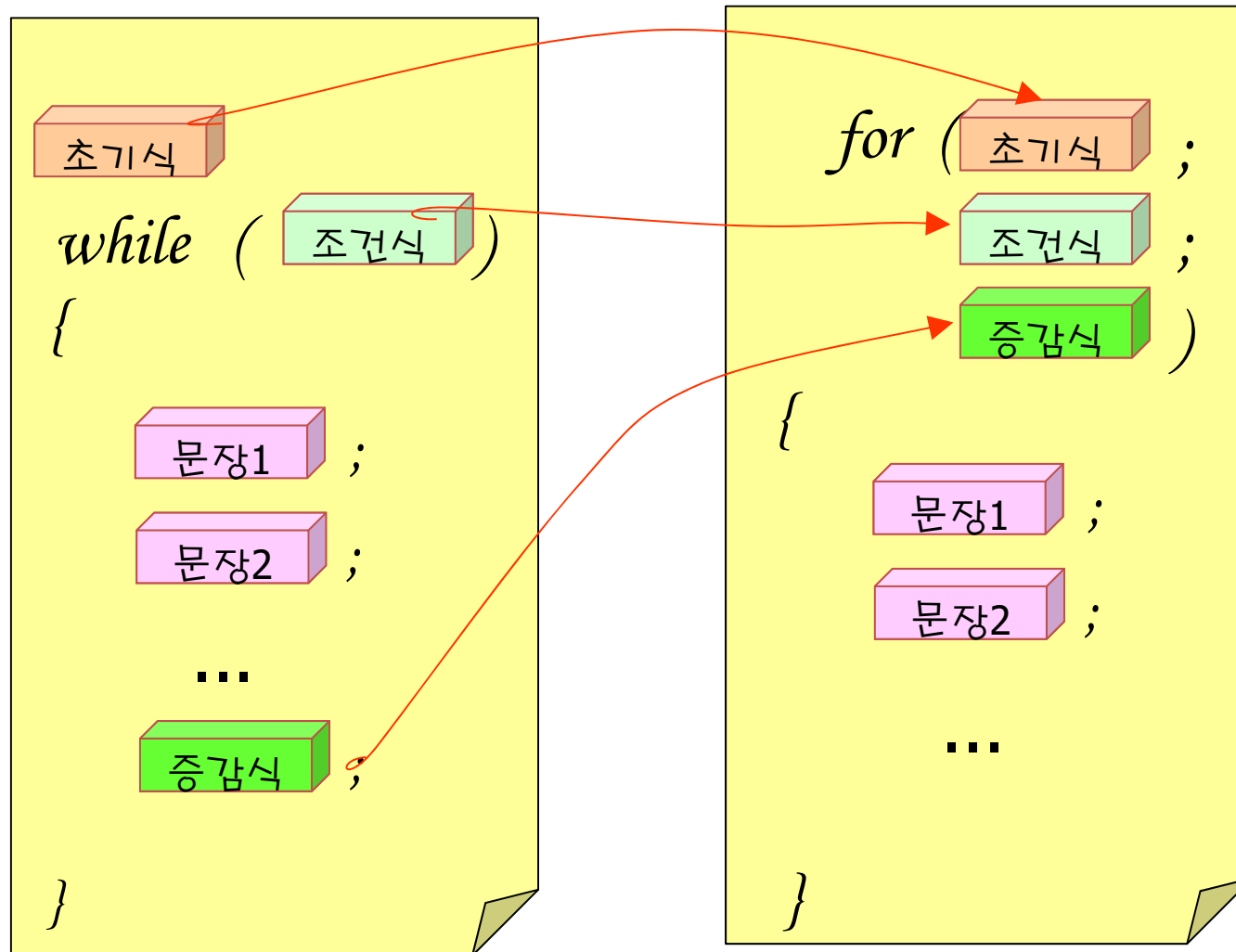
```
    → for(i = 1; i <= n; i++)  
        printf("%5d   %5d\n", i, i * i * i);
```

```
    return 0;
```

```
}
```



# while 루프와 for 루프와의 관계



# 팩토리얼 계산 예제(for-loop)

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

int main(void)
{
    double fact = 1.0;
    int n;
    printf("정수를 입력하시오: ");
    scanf("%d", &n);
    for (int i = 1; i <= n; i++)
    {
        fact = fact * i;
    }
    printf("%d!은 %f입니다.", n, fact);
    return 0;
}
```

```
for (int i = n; i >= 1; i--)
{
    fact = fact * i;
}
```



# 다양한 증감수식의 형태

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

뺄셈 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");
```

2배씩 증가

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

어떤 수식이라도 가능



# 다양한 for 루프

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

한 부분이 없을 수도 있다.  
(초기값 설정을 다른 곳에서 실행)

```
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

```
for ( ; ; )  
    printf("Hello World!\n");
```

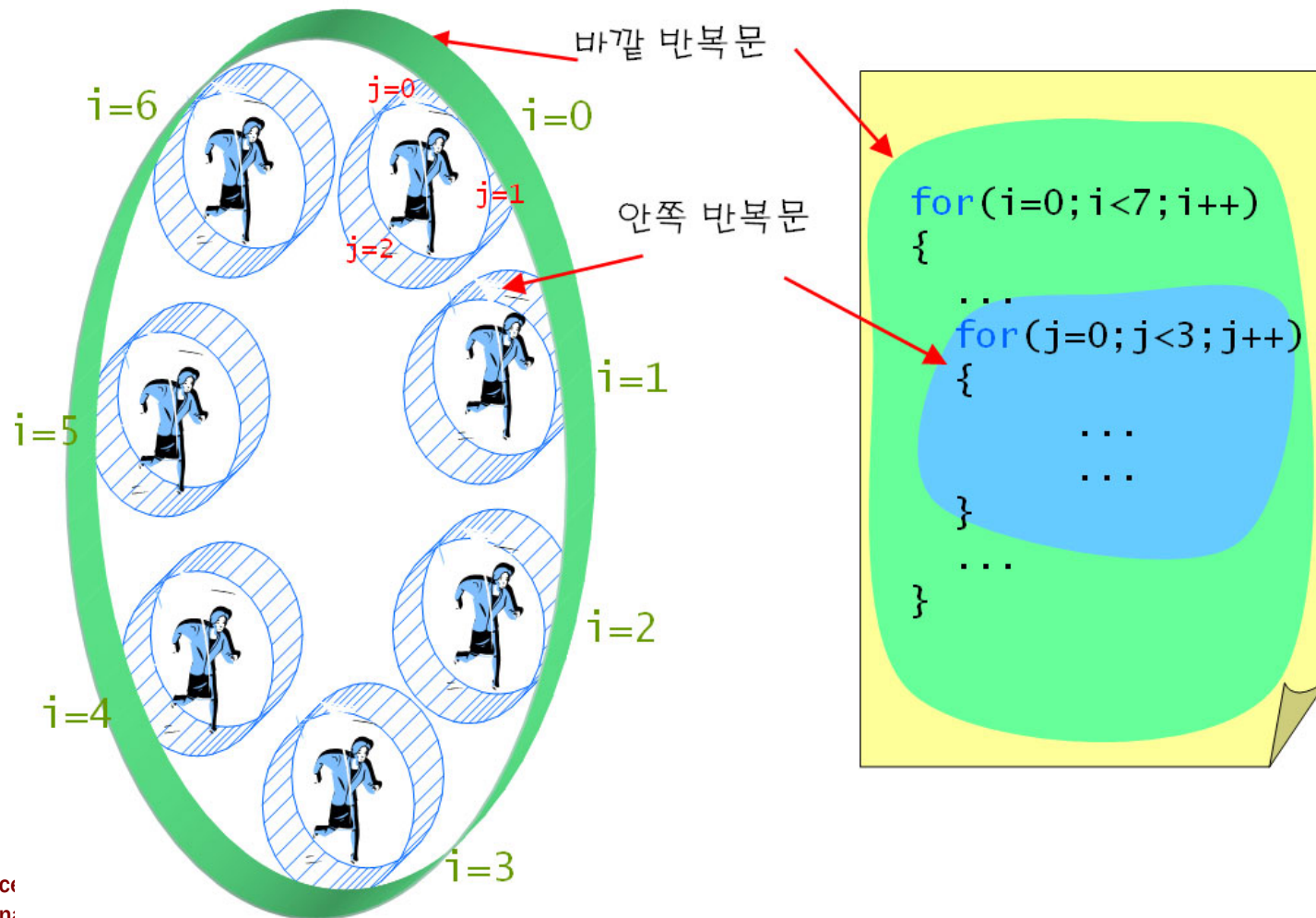
무한 반복 루프



## **중첩 반복문 (Nested Loop)**

# 중첩 반복문 (nested loop)

◆ 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++)
```

②

```
{
```

```
    for(x = 0; x < 10; x++)
```

①

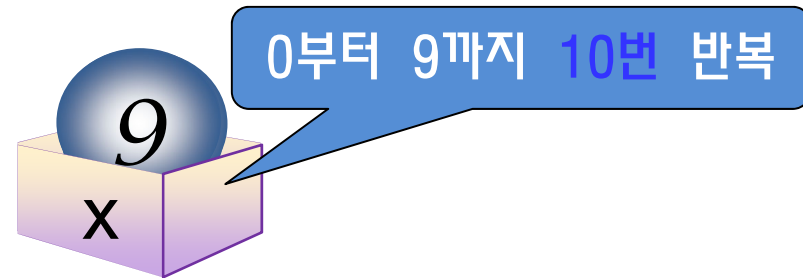
```
        printf(" * ");
```

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

```
    }
```

```
    return 0;
```

```
}
```



# 예제

// break를 이용하여 무한루프를 탈출한다.

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

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

```
int main(void)
```

```
{
```

```
    double v;
```

```
    while(1)
```

```
    {
```

```
        printf("실수 값을 입력하시오: ");
```

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

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

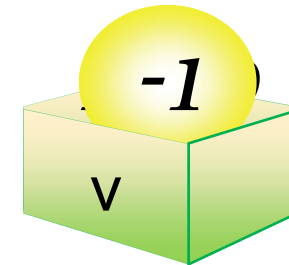
```
            break;
```

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

```
    }
```

```
    return 0;
```

```
}
```



실수 값을 입력하시오: 9.0  
9.000000의 제곱근은 3.000000입니다.  
실수 값을 입력하시오: 12.0  
12.000000의 제곱근은 3.464102입니다.  
실수 값을 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;
```

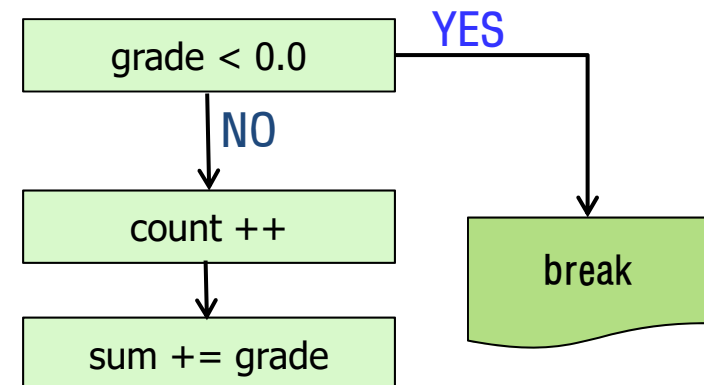
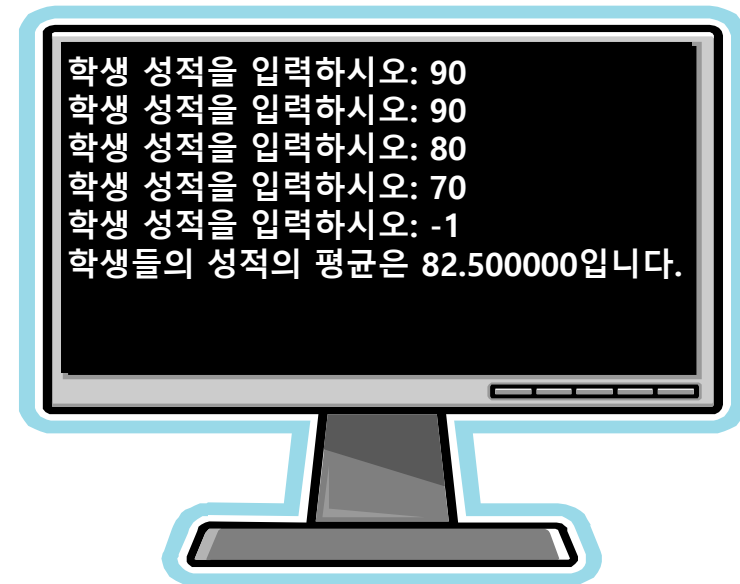
```
    }
```

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

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

```
    return 0;
```

음수 입력 시  
while문을 나옴



# 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"); // 내부 반복문이 종료될 때마다 실행
    }
    return 0;
}
```



# 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;
```

```
}
```

OUT 으로 goto



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



# 예제

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

```
int main(void)
```

```
{
```

```
    int i;
```

```
    for(i=0 ; i<10 ; i++)
```

```
    {
```

```
        if( i%3 == 0 )
```

```
            continue;
```

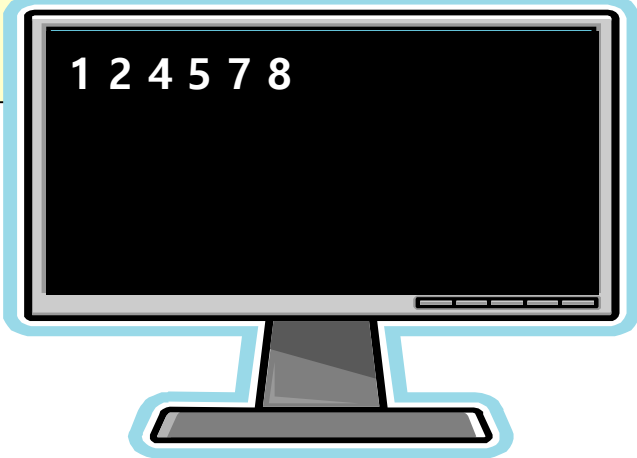
```
        printf("%d ", i);
```

```
    }
```

```
    return 0;
```

```
}
```

3의 배수 는 건너뛴다.

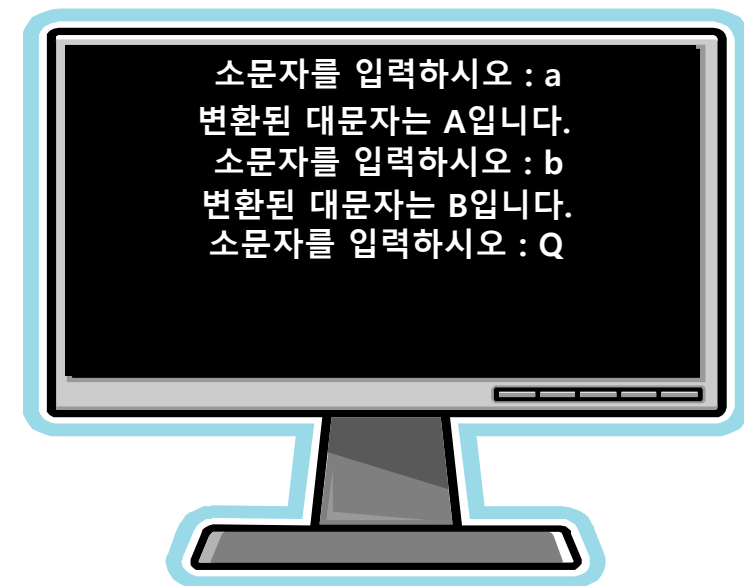
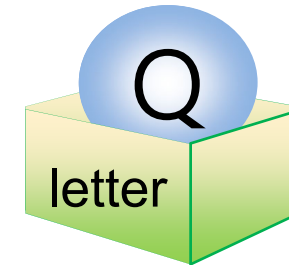


1 2 4 5 7 8



# 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;  
}
```



# 반복문을 사용한 달력 출력

```
#include <stdio.h>
enum WEEKDAY {SUN, MON, TUE, WED, THR, FRI, SAT};
#define START_DAY WED // 첫 번째 날이 수요일
#define DAYS_OF_MONTH 31 // 달의 일수

int main(void)
{
    int weekday, date;
    printf("=====\n");
    printf("일 월 화 수 목 금 토\n");
    printf("=====\n");
    // 월요일부터 수요일까지
    for(weekday = 0; weekday < START_DAY ; weekday++)
        printf(" "); // 1일의 요일 위치까지 공백 출력
    for(date = 1; date <= DAYS_OF_MONTH ; date++)
    {
        if( weekday == SUN )
        {
            weekday = 0; // 일요일이면 줄바꿈을 출력
            printf("\n");
        }
        weekday = (weekday+1) % 7;
        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
=====
```

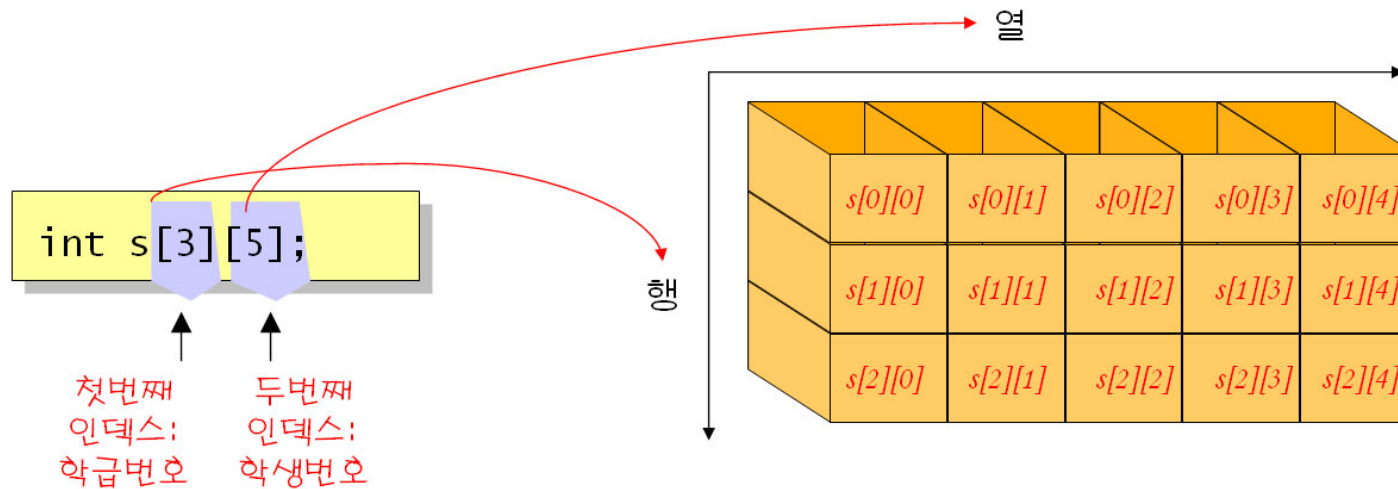




## 다차원 배열 기반 행렬계산

# 2차원 배열

```
int s[10];    // 1차원 배열  
int s[3][10]; // 2차원 배열  
int s[5][3][10]; // 3차원 배열
```



# 2차원 배열의 활용

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

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

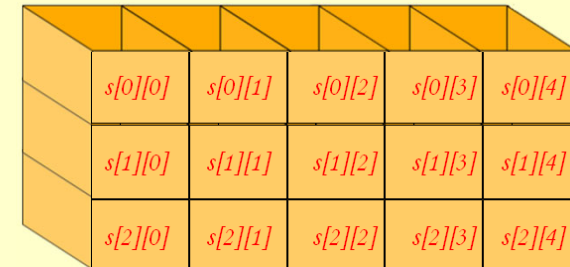
```
    int s[3][5];        // 2차원 배열 선언  
    int i, j;           // 2개의 인덱스 변수  
    int value = 0;      // 배열 원소에 저장되는 값
```

```
    for(i=0;i<3;i++)  
        for(j=0;j<5;j++)  
            s[i][j] = value++;
```

```
    for(i=0;i<3;i++)  
        for(j=0;j<5;j++)  
            printf("%d\n", s[i][j]);
```

```
    return 0;
```

```
}
```



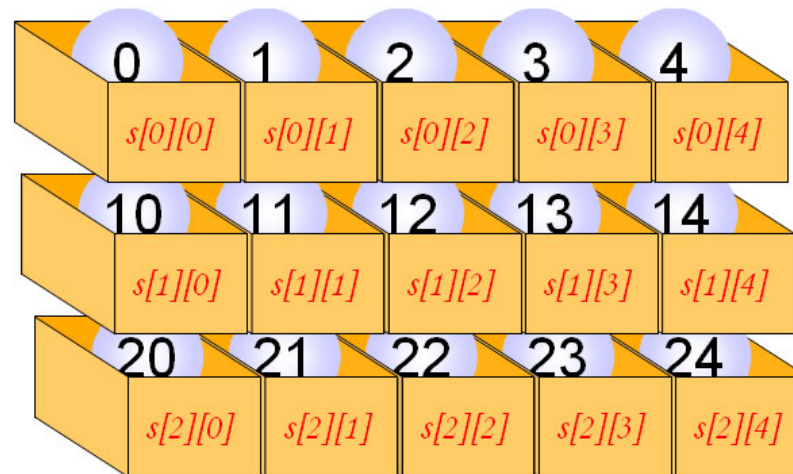
A 3D diagram representing a 3x5 array. It consists of three rows and five columns of orange boxes. Each box contains a red text label indicating its memory address. The labels are: Row 0: s[0][0], s[0][1], s[0][2], s[0][3], s[0][4]; Row 1: s[1][0], s[1][1], s[1][2], s[1][3], s[1][4]; Row 2: s[2][0], s[2][1], s[2][2], s[2][3], s[2][4].

s[0][0]	s[0][1]	s[0][2]	s[0][3]	s[0][4]
s[1][0]	s[1][1]	s[1][2]	s[1][3]	s[1][4]
s[2][0]	s[2][1]	s[2][2]	s[2][3]	s[2][4]



## 2차원 배열의 초기화

```
int s[3][5] = {  
    { 0, 1, 2, 3, 4}, // 0 번째 행의 원소들의 초기값  
    { 10, 11, 12, 13, 14}, // 1 번째 행의 원소들의 초기값  
    { 20, 21, 22, 23, 24} // 2 번째 행의 원소들의 초기값  
};
```



# 3차원 배열

`int s [6][3][5];`

첫번째    두번째    세번째  
인덱스:    인덱스:    인덱스:  
학년번호    학급번호    학생번호

```
#include <stdio.h>
int main(void)
{
    int s[3][3][3];    // 3차원 배열 선언
    int x, y, z;       // 3개의 인덱스 변수
    int i = 1;         // 배열 원소에 저장되는 값

    for(z=0;z<3;z++)
        for(y=0;y<3;y++)
            for(x=0;x<3;x++)
                s[z][y][x] = i++;

    return 0;
}
```



# 다차원 배열 인수

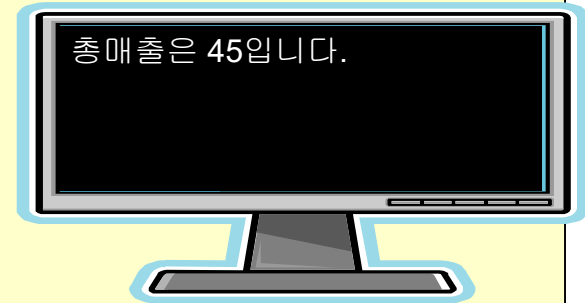
```
#include <stdio.h>
#define YEARS    3
#define PRODUCTS 5

int sum(int grade[][PRODUCTS]);

int main(void)
{
    int sales[YEARS][PRODUCTS] = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };
    int total_sale;
    total_sale = sum(sales);

    printf("총매출은 %d입니다.\n", total_sale);
    return 0;
}

int sum(int grade[][PRODUCTS])
{
    int y, p;
    int total = 0;
    for(y = 0; y < YEARS; y++)
        for(p = 0; p < PRODUCTS; p++)
            total += grade[y][p];
    return total;
}
```



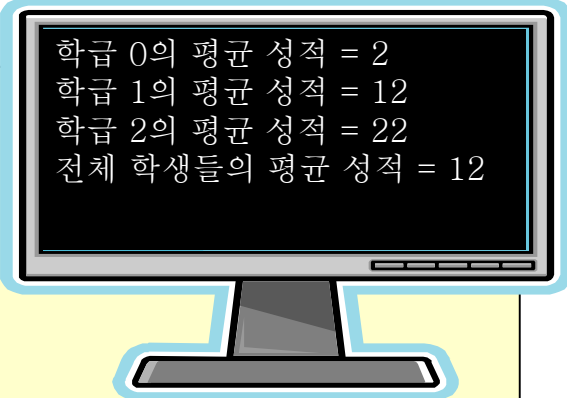
첫번째 인덱스의 크기는  
적지 않아도 된다.



# 다차원 배열 예제

```
#include <stdio.h>
#define CLASSES 3
#define STUDENTS 5

int main(void)
{
    int s[CLASSES][STUDENTS] = {
        { 0, 1, 2, 3, 4 },    // 0 번째 행의 원소들의 초기값
        { 10, 11, 12, 13, 14 }, // 1 번째 행의 원소들의 초기값
        { 20, 21, 22, 23, 24 }, // 2 번째 행의 원소들의 초기값
    };
    int clas, student, total, subtotal;
    total = 0;
    for(clas = 0; clas < CLASSES; clas++)
    {
        subtotal = 0;
        for(student = 0; student < STUDENTS; student++)
            subtotal += s[clas][student];
        printf("학급 %d의 평균 성적= %d\n", clas, subtotal / STUDENTS);
        total += subtotal;
    }
    printf("전체 학생들의 평균 성적= %d\n", total / (CLASSES * STUDENTS));
    return 0;
}
```



학급 0의 평균 성적 = 2  
학급 1의 평균 성적 = 12  
학급 2의 평균 성적 = 22  
전체 학생들의 평균 성적 = 12

## 2차원 배열과 행렬 (Matrix)

- ◆ 행렬(matrix)은 자연과학에서 많은 문제를 해결하는데 사용

$$A = \begin{bmatrix} 2 & 3 & 0 \\ 8 & 9 & 1 \\ 7 & 0 & 5 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 & 7 & 0 & 0 \\ 9 & 0 & 0 & 0 & 0 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & 0 & 0 & 0 \end{bmatrix}$$

Mathematics - ELEMENTARY MATRIX OPERATIONS

OPERATION: MULTIPLY EACH ELEMENT IN 2<sup>nd</sup> Row by 7:

$A = \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{bmatrix}$

1) FIND  $E$   $2 \times 2$   $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 0 \\ 0 & 7 \end{bmatrix}$   
 $I$   $E$

2) PREMULT  $\begin{bmatrix} 1 & 0 \\ 0 & 7 \end{bmatrix} \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{bmatrix} \Rightarrow \begin{bmatrix} 1(0)+0(3) & 1(1)+0(4) & 1(2)+0(5) \\ 0(0)+7(3) & 0(1)+7(4) & 0(2)+7(5) \end{bmatrix}$





## 2차원 배열을 이용한 행렬의 표현

```
#include <stdio.h>
#define ROWS 3
#define COLS 3

int main(void)
{
    int A[ROWS][COLS] = { { 2,3,0 },
                           { 8,9,1 },
                           { 7,0,5 } };
    int B[ROWS][COLS] = { { 1,0,0 },
                           { 1,0,0 },
                           { 1,0,0 } };

    int C[ROWS][COLS];
    int r, c; // row, column
    // 두 개의 행렬을 더한다.
    for(r = 0; r < ROWS; r++)
        for(c = 0; c < COLS; c++)
            C[r][c] = A[r][c] + B[r][c];
    // 행렬을 출력한다.
    for(r = 0; r < ROWS; r++)
    {
        for(c = 0; c < COLS; c++)
            printf("%d ", C[r][c]);
        printf("\n");
    }
    return 0;
}
```

중첩 for 루프를 이용하여 행렬 A의 각  
원소들과 행렬의 B의 각 원소들을 서  
로 더하여 행렬 C에 대입한다.



# 4x4 행렬의 계산

```
/** matrix4_4.cpp  */

#include <iostream>
#include <iomanip>
using namespace std;

#define SIZE_4 4

void printMtrx(int mA[][SIZE_4], int size_n);
void addMtrx(int mA[][SIZE_4], int mB[][SIZE_4], int mX[][SIZE_4], int size_n);
void subtractMtrx(int mA[][SIZE_4], int mB[][SIZE_4], int mX[][SIZE_4], int size_n);
void multiplyMtrx(int mA[][SIZE_4], int mB[][SIZE_4], int mX[][SIZE_4], int size_n);

int main()
{
    int mA[4][4] = {{1, 2, 3, 4},
                    {5, 6, 7, 8},
                    {9, 10, 11, 12},
                    {13, 14, 15, 16}};
    int mB[4][4] = {{1, 0, 0, 0},
                    {0, 2, 0, 0},
                    {0, 0, 3, 0},
                    {0, 0, 0, 4}};
    int mC[4][4];
    int mD[4][4];
    int mE[4][4];
}
```

Matrix mA:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Matrix mB:

1	0	0	0
0	2	0	0
0	0	3	0
0	0	0	4

e

Pror. Young-Iak Kim



## 4x4 행렬의 계산

```
/** matrix4_4.cpp (cont.) */

    printf("\n Matrix mA:\n");
    printMtrx(mA, 4);

    printf("\n Matrix mB:\n");
    printMtrx(mB, 4);

    addMtrx(mA, mB, mC, 4);
    printf("\n Matrix mC = mA + mB:\n");
    printMtrx(mC, 4);

    subtractMtrx(mA, mB, mD, 4);
    printf("\n Matrix mD = mA - mB:\n");
    printMtrx(mD, 4);

    multiplyMtrx(mA, mB, mE, 4);
    printf("\n Matrix mE = mA x mB:\n");
    printMtrx(mE, 4);

    printf("\n");

    return 0;

}
```



# 확장 완성형 코드를 사용한 4x4 행렬의 출력

```
void printMtrx(int mA[][SIZE_4], int size)
{
    unsigned char a6 = 0xA6, a1 = 0xA1, a2 = 0xA2;
    unsigned char a3 = 0xA3, a4 = 0xA4, a5 = 0xA5;

    for (int i=0; i< size_n; i++) {
        for (int j=0; j< size; j++)
        {
            if ((i==0) && (j==0))
                printf("%c%c%c%3d", a6, a3, mA[i][j]);
            else if ((i==0) && j==(size -1))
                printf("%3d%c%c", mA[i][j], a6, a4);
            else if ((i>0) && (i<size-1) && (j==0))
                printf("%c%c%c%3d", a6, a2, mA[i][j]);
            else if ((i>0) && (i<size-1) && (j== (size -1)))
                printf("%3d%c%c", mA[i][j], a6, a2);
            else if ((i==(size-1)) && (j==0))
                printf("%c%c%c%3d", a6, a6, mA[i][j]);
            else if ((i==(size-1)) && (j==(size -1)))
                printf("%3d%c%c", mA[i][j], a6, a5);
            else
                printf("%3d", mA[i][j]);
        }
        printf("\n");
    }
}
```

출력 결과	확장 완성형 코드
—	0xa6, 0xa1
	0xa6, 0xa2
┌	0xa6, 0xa3
┐	0xa6, 0xa4
└	0xa6, 0xa5
┘	0xa6, 0xa6

**Matrix mA:**

```
[ 1  2  3  4
  5  6  7  8
  9 10 11 12
 13 14 15 16]
```

**Matrix mB:**

```
[ 1  0  0  0
  0  2  0  0
  0  0  3  0
  0  0  0  4]
```



# 행렬의 곱셈

## ◆ 행렬의 곱셈 계산

$$\begin{array}{|c|c|c|} \hline \boxed{x_{00} = a_{00}b_{00} + a_{01}b_{10} + a_{02}b_{20}} & \boxed{x_{01} = a_{00}b_{01} + a_{01}b_{11} + a_{02}b_{21}} & \boxed{x_{02} = a_{00}b_{02} + a_{01}b_{12} + a_{02}b_{22}} \\ \hline \boxed{x_{10} = a_{10}b_{00} + a_{11}b_{10} + a_{12}b_{20}} & \boxed{x_{11} = a_{10}b_{01} + a_{11}b_{11} + a_{12}b_{21}} & \boxed{x_{12} = a_{10}b_{02} + a_{11}b_{12} + a_{12}b_{22}} \\ \hline \boxed{x_{20} = a_{20}b_{00} + a_{21}b_{10} + a_{22}b_{20}} & \boxed{x_{21} = a_{20}b_{01} + a_{21}b_{11} + a_{22}b_{21}} & \boxed{x_{22} = a_{20}b_{02} + a_{21}b_{12} + a_{22}b_{22}} \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline a_{00} & a_{01} & a_{02} \\ \hline a_{10} & a_{11} & a_{12} \\ \hline a_{20} & a_{21} & a_{22} \\ \hline \end{array} \times \begin{array}{|c|c|c|} \hline b_{00} & b_{01} & b_{02} \\ \hline b_{10} & b_{11} & b_{12} \\ \hline b_{20} & b_{21} & b_{22} \\ \hline \end{array}$$

# 4x4 행렬의 덧셈, 뺄셈, 곱셈

```
void addMtrx(int mA[][SIZE_4], int mB[][SIZE_4], int mX[][SIZE_4], int size)
```

```
{  
    for (int i=0; i<size; i++)  
        for (int j=0; j<size; j++)  
            mX[i][j] = mA[i][j] + mB[i][j];  
}
```

```
void subtractMtrx(int mA[][SIZE_4], int mB[][SIZE_4], int mX[][SIZE_4], int size)
```

```
{  
    for (int i=0; i<size; i++)  
        for (int j=0; j<size; j++)  
            mX[i][j] = mA[i][j] - mB[i][j];  
}
```

```
void multiplyMtrx(int mA[][SIZE_4], int mB[][SIZE_4], int mX[][SIZE_4], int size)
```

```
{  
    for (int i=0; i<size; i++)  
        for (int j=0; j<size; j++)  
        {  
            mX[i][j] = 0;  
            for (int k=0; k<size; k++)  
                mX[i][j] += mA[i][k] * mB[k][j];  
        }  
}
```



# 실행결과

Matrix mA:

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Matrix mB:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

Matrix mC = mA + mB:

$$\begin{bmatrix} 2 & 2 & 3 & 4 \\ 5 & 8 & 7 & 8 \\ 9 & 10 & 14 & 12 \\ 13 & 14 & 15 & 20 \end{bmatrix}$$

Matrix mD = mA - mB:

$$\begin{bmatrix} 0 & 2 & 3 & 4 \\ 5 & 4 & 7 & 8 \\ 9 & 10 & 8 & 12 \\ 13 & 14 & 15 & 12 \end{bmatrix}$$

Matrix mE = mA x mB:

$$\begin{bmatrix} 1 & 4 & 9 & 16 \\ 5 & 12 & 21 & 32 \\ 9 & 20 & 33 & 48 \\ 13 & 28 & 45 & 64 \end{bmatrix}$$



## **Homework 3**



## Homework 3

**3.1 표준입력장치 (키보드)에서 문자 1개를 입력 받은 후, 이 문자가 대문자, 소문자, 모음 (vowel: a, e, i, o, u), 자음 (consonant), 숫자, 기호인지 구분하고, 그 결과를 출력하는 알고리즘을 pseudo code로 작성하고, 이에 대한 C 프로그램을 작성하라. (ASCII 표를 참조할 것.)** 보고서에 주석문이 포함된 프로그램 소스코드는 글상자로 포함시키고, 실행 결과는 capture 하여 포함시킬 것.

**3.2 연도 (year)을 입력받고, 이 연도의 1월 ~ 12월 달력을 월별로 출력하는 알고리즘의 pseudo code를 작성하라.** 해당 연도가 윤년인가를 확인하여, 이에 따라 정확한 2월 출력이 될 수 있게 할 것. 각 달의 달력에서는 요일이 표시되어야 하며, 1주일 단위로 줄 바꿈이 표시되어야 한다. 이 알고리즘을 C 프로그램으로 작성하고, 주석문이 포함된 소스코드는 글상자로 보고서에 포함시키며, 실행결과 화면은 캡처하여 보고서에 포함시킬 것.



### 3.3 중첩 반복문을 사용하여 12 x 12 곱셈표를 출력하는 알고리즘의 pseudo code를 작성하라.

이 프로그램은 아래에서 보는 형식으로 12 x 12 곱셈표를 출력하여야 한다. 출력되는 값은 5자리씩을 가지며, 오른쪽 맞춤으로 printf()의 출력 포맷을 설정할 것.

이 알고리즘을 C 프로그램으로 작성하고, 주석문이 포함된 소스코드는 글상자로 포함하며, 실행결과 화면은 캡처하여 보고서로 제출하라.

```
>>> 12 x 12 Multiplication Table <<<
  | 1  2  3  4  5  6  7  8  9 10 11 12
--+--+--+--+--+--+--+--+--+--+--+--+
1 | 1  2  3  4  5  6  7  8  9 10 11 12
2 | 2  4  6  8 10 12 14 16 18 20 22 24
3 | 3  6  9 12 15 18 21 24 27 30 33 36
4 | 4  8 12 16 20 24 28 32 36 40 44 48
5 | 5 10 15 20 25 30 35 40 45 50 55 60
6 | 6 12 18 24 30 36 42 48 54 60 66 72
7 | 7 14 21 28 35 42 49 56 63 70 77 84
8 | 8 16 24 32 40 48 56 64 72 80 88 96
9 | 9 18 27 36 45 54 63 72 81 90 99 108
10 | 10 20 30 40 50 60 70 80 90 100 110 120
11 | 11 22 33 44 55 66 77 88 99 110 121 132
12 | 12 24 36 48 60 72 84 96 108 120 132 144

계속하려면 아무 키나 누르십시오 . . .
```

