



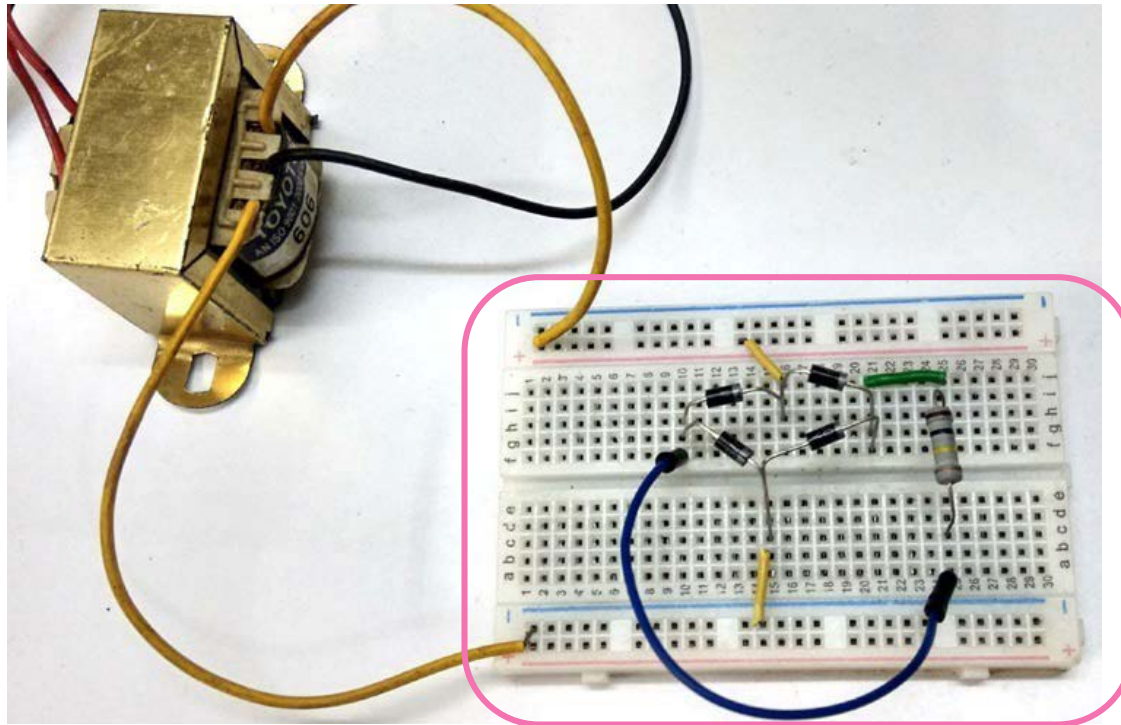
C Programming

제10장 배열

한밭대학교 정보통신공학과
최 해 철

Review – main()

- _____함수는 **빵판(breadboard)**이다!



Review – 함수(function)

알고리즘은 _____ 다!



```
main()
{
    ...
}
```

Recipe

재료 준비

함수 1

함수 2

출력

함수 1

함수 2



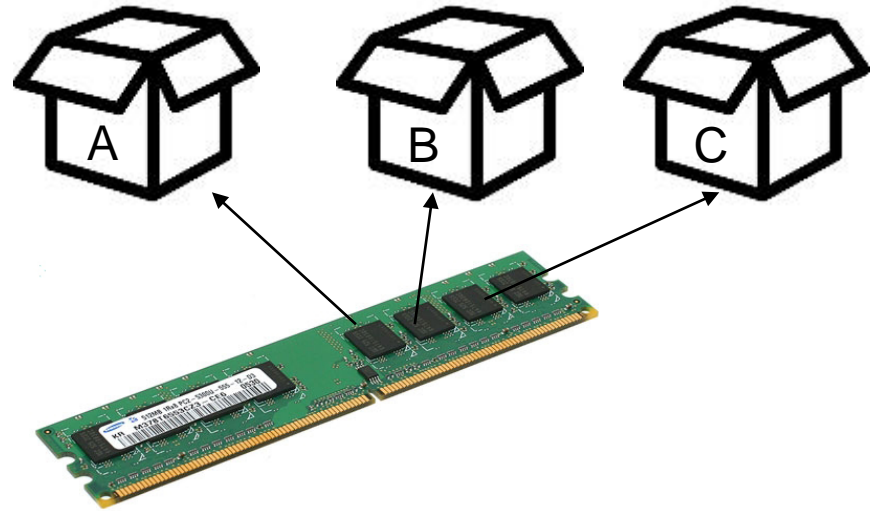
Review- 변수

- 4 x 587 ?



- □ x □ ?

$$\rightarrow C = A \times B$$



```
int A, B, C;  
A=4; B=587;  
C = A*B;
```

```
// 1. _____  
// 2. _____  
// 3. _____
```

Review – 프로그램 작성 기본 구조

- 알고리즘은 **절차**다!
- 1) 프로그램 짤 준비 → main()함수 만들기
 - 빵판 준비, 조리실 준비

```
#include <stdio.h>

int main ()
{

    return 0;
}
```

Review – 프로그램 작성 기본 구조

- 알고리즘은 **절차**다!
- 2) 데이터 저장할 공간 준비 → 변수 선언 (int, float, char)
 - 재료를 담을 냄비 준비

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

```
int main ()
```

```
{
```

```
    int x, y, z;
```

```
    return 0;
```

```
}
```

Review – 프로그램 작성 기본 구조

- 알고리즘은 **절차**다!
- 3) 데이터 입력 받기 → 입력 부분 작성 (대입, scanf() 활용)
 - 재료를 냄비에 담기

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

```
int main ()
```

```
{
```

```
    int x, y, z;
```

```
    x = 4;
```

```
    scanf("%d", y);
```

```
    return 0;
```

```
}
```

Review – 프로그램 작성 기본 구조

- 알고리즘은 **절차**다!
- 4) 데이터 처리 → 연산(반복문, 조건문, 함수 활용)
 - 넵비 꿔이기, 재료 볶기...

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

```
int main ()
```

```
{
```

```
    int x, y, z;
```

```
    x = 4;
```

```
    scanf("%d", y);
```

```
    z = x*y;
```

```
    return 0;
```

```
}
```


Review – 프로그램 작성 기본 구조

- 알고리즘은 **절차**다!
- 4) 결과 출력 (printf() 활용)
 - 요리 내놓기, 화면에 보이기

```
#include <stdio.h>

int main ()
{
    int x, y, z;
    x = 4;
    scanf("%d", y);
    z = x*y;
    printf("%d", z);
    return 0;
}
```

Review – 프로그램 작성 기본 구조

- 알고리즘은 **절차**다!

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

1. _____

```
int main ()
```

```
{
```

```
    int x, y, z;
```

2. _____

```
    x = 4;
```

```
    scanf("%d", y);
```

3. _____

```
    z = x*y;
```

4. _____

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

5. _____

```
    return 0;
```

```
}
```

이번 장에서 학습할 내용



- 배열의 개념
- 배열의 선언과 초기화
- 일차원 배열

- 배열과 함수
- 다차원 배열

배열을 사용하면
한 번에 여러
개의 값을 저장할
수 있는 공간을
할당받을 수
있다.



배열의 필요성



int x

배열의 필요성



```
int x1;  
int x2;  
int x3;  
int x4;  
int x5;  
int x6;
```

배열의 필요성



```
int x[6];
```

배열의 필요성

- 학생이 10명이 있고 이들의 평균 성적을 계산한다고 가정하자,

개별 변수를 사용
하는 방법은 학생
수가 많아지면 번
거로워집니다..



방법 #1: 개별 변수 사용

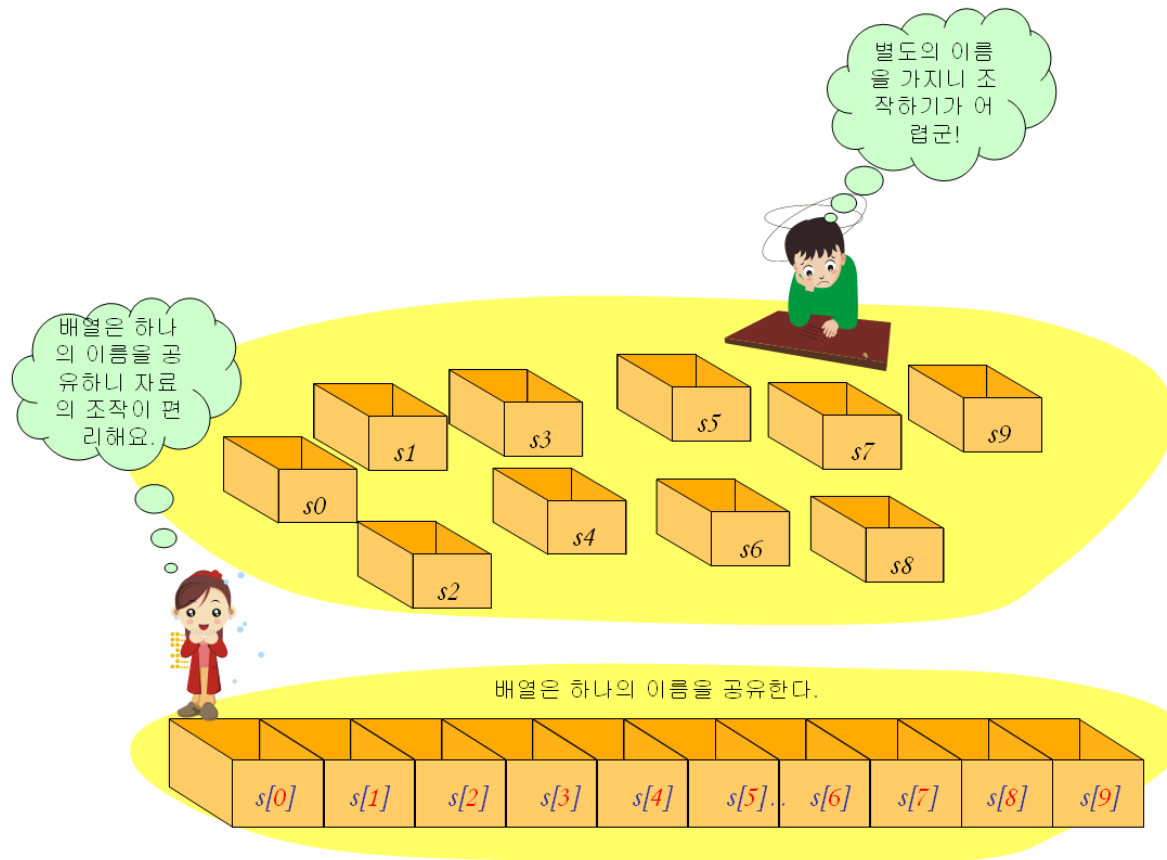
```
int s0;  
int s1;  
...  
int s9;
```

방법 #2: 배열 사용

```
int s[10];
```

배열이란?

- 배열(array): _____의 데이터가 _____ 저장되어 있는 데이터 저장 장소
- 배열을 이용하면 여러 개의 값을 _____으로 처리할 수 있다.



배열이란?

- 배열(array): **동일한 타입**의 데이터가 여러 개 저장되어 있는 데이터 저장 장소
 - 섞어 담을 수는 없다!!



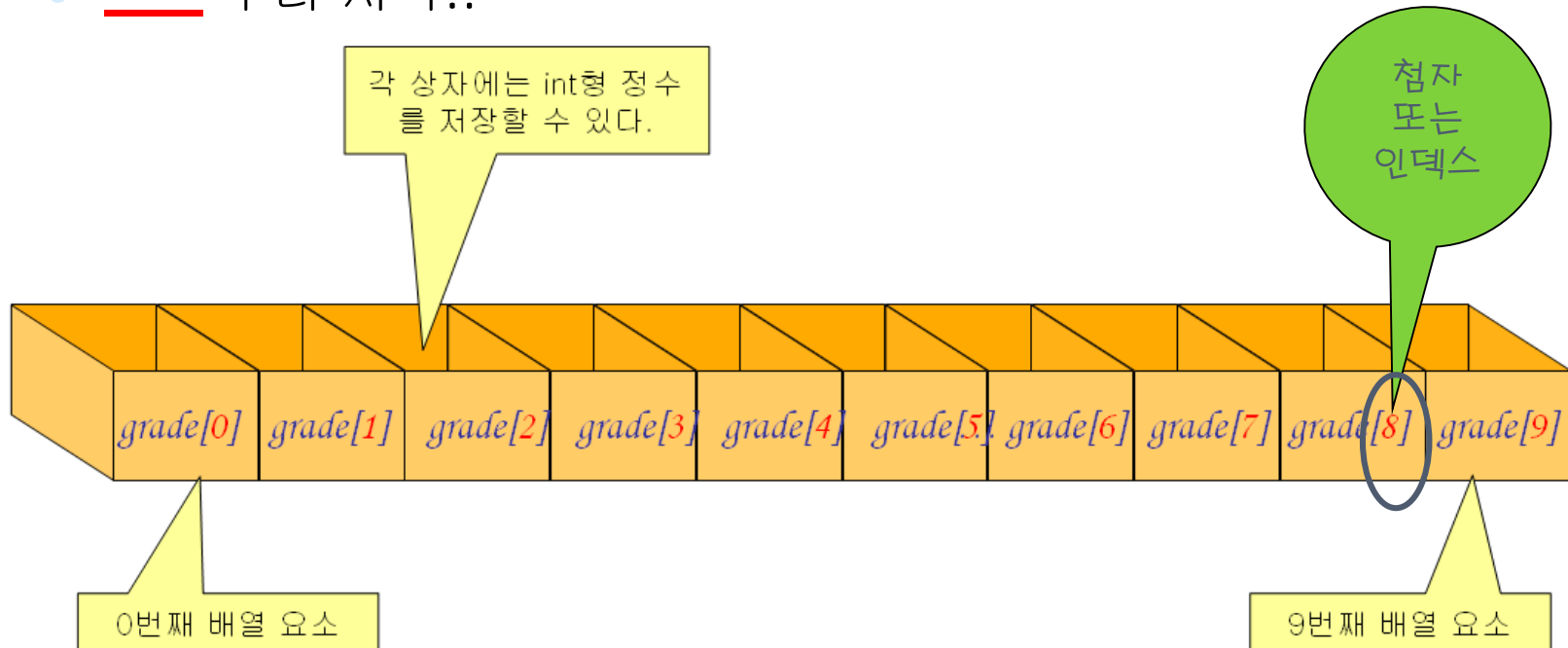
```
int x[6];
```



```
char y[6];
```

배열 원소와 인덱스

- **인덱스(index):** 배열 원소의 번호
 - 배열 안에 들어있는 각각의 데이터들은 로 되어 있는 번호(첨자)에 의하여 접근
 - 부터 시작!!



배열의 선언

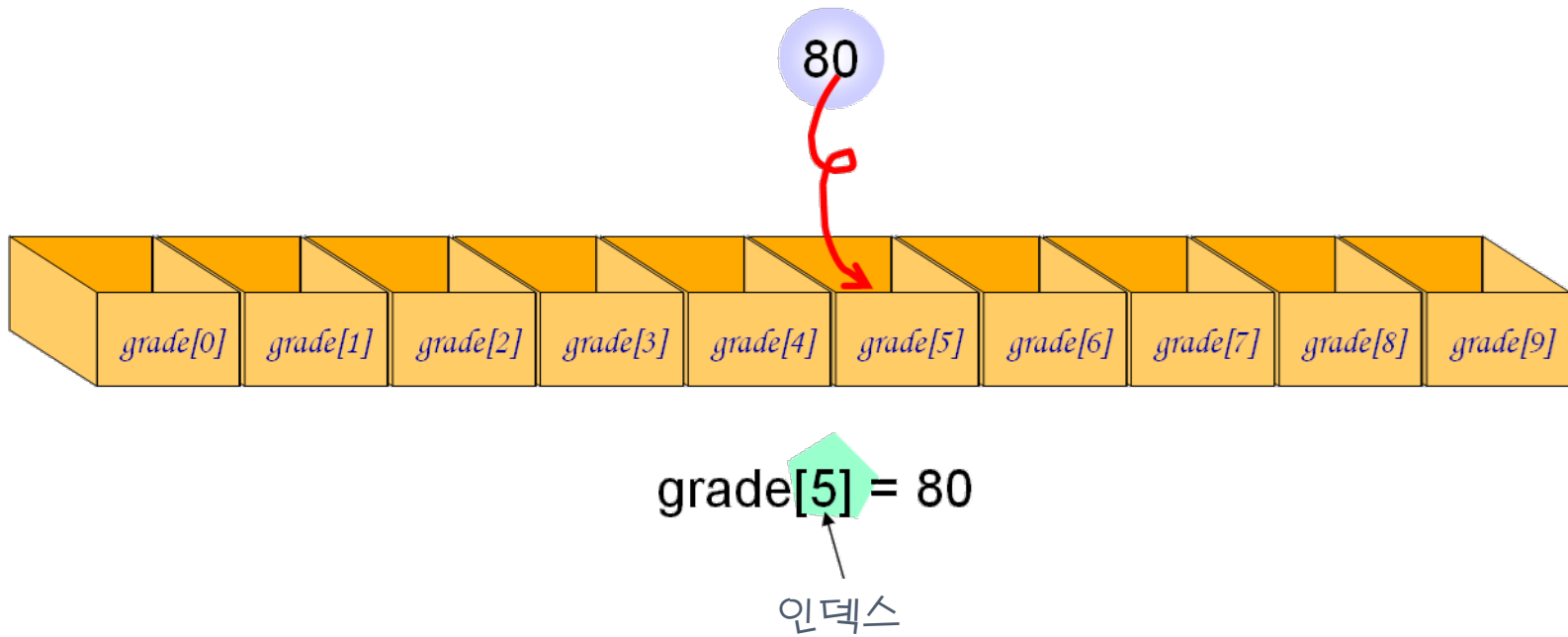
Syntax: 배열 선언



- 자료형: 배열 원소들이 `int`형라는 것을 의미
- 배열 이름: 배열을 사용할 때 사용하는 이름이 `grade`
(배열이 저장되는 메모리의 주소 값)
- 요소의 개수: 배열 원소의 개수가 10개
- **인덱스(배열 번호)는 항상 0부터 시작한다.**

```
int score[60];           // 60개의 int형 값을 가지는 배열 grade
float cost[12];          // 12개의 float형 값을 가지는 배열 cost
char name[50];           // 50개의 char형 값을 가지는 배열 name
char src[10], dst[10];   // 2개의 문자형 배열을 동시에 선언
int index, days[7];      // 일반 변수와 배열을 동시에 선언
```

배열 원소 접근



```
grade[5] = 80;  
grade[1] = grade[0];  
grade[i] = 100;      // i는 정수 변수  
grade[i+2] = 100;    // 수식이 인덱스가 된다.  
grade[index[3]] = 100; // index[]는 정수 배열
```

배열 선언 예제

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

```
int main(void)
```

```
{
```

```
    int i;
```

```
    int grade[5];
```

```
    grade[0] = 10;
```

```
    grade[1] = 20;
```

```
    grade[2] = 30;
```

```
    grade[3] = 40;
```

```
    grade[4] = 50;
```

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

```
        printf("grade[%d]=%d\n", i, grade[i]);
```

```
    return 0;
```

```
}
```



```
grade[0]=10  
grade[1]=20  
grade[2]=30  
grade[3]=40  
grade[4]=50
```

배열과 반복문

- 배열의 가장 큰 장점은 _____을 사용하여서 배열의 원소를 간편하게 처리할 수 있다는 점
- 인덱스에 변수 사용 가능



```
grade[0] = 0;  
grade[1] = 0;  
grade[2] = 0;  
grade[3] = 0;  
grade[4] = 0;
```



```
#define SIZE 5  
...  
for(i=0 ; i<SIZE ; i++)  
    grade[i] = 0;
```

잘못된 인덱스 문제

- 인덱스가 _____ 프로그램에 치명적인 오류 발생

```
int grade[5];  
...  
grade[5] = 60;    // 치명적인 오류!
```

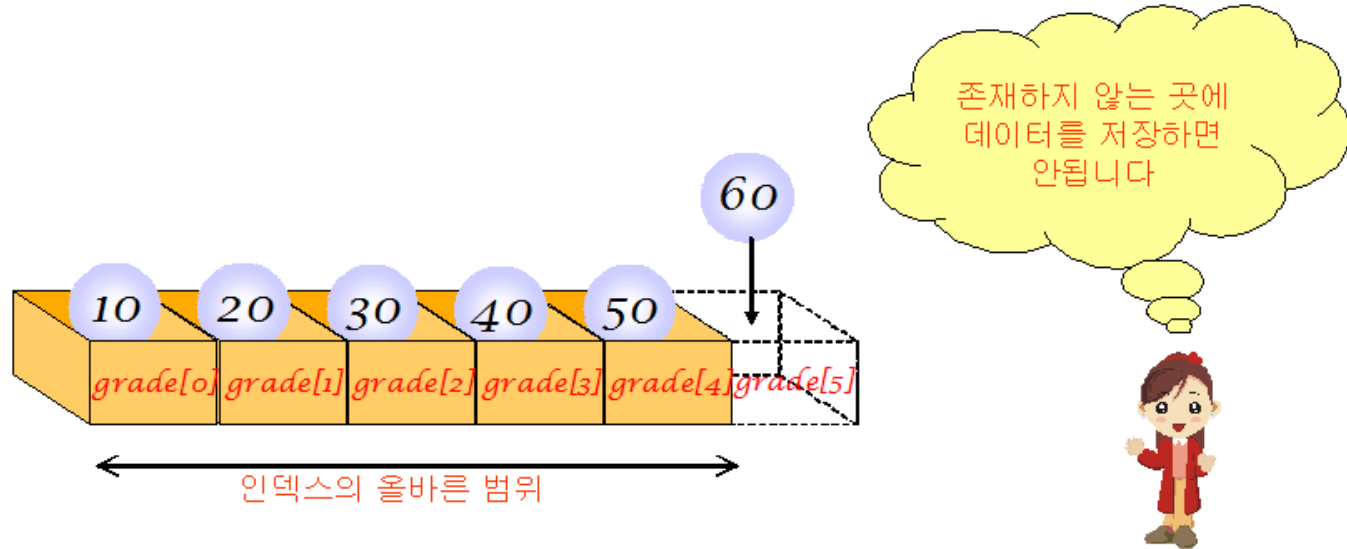


그림 8.4 잘못된 인덱스

잘못된 인덱스로 접근하는 경우



```
#include <stdio.h>
#define SIZE 5

int main(void)
{
    int array[SIZE] = {1, 2, 3, 4, 5};
    int i;

    for(i = 0; i <= SIZE; i++)
        printf("array[%d] %d\\ n", i, array[i]);

    return 0;
}
```



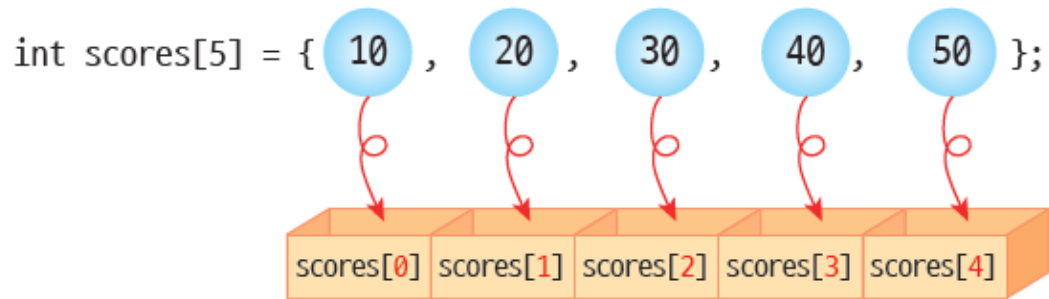
array[0]	1
array[1]	2
array[2]	3
array[3]	4
array[4]	5
array[5]	1245120

중간 점검

1. n 개의 원소를 가지는 배열의 경우, 첫 번째 원소의 번호는 무엇인가?
2. n 개의 원소를 가지는 배열의 경우, 마지막 원소의 번호는 무엇인가?
3. 배열 원소의 번호 혹은 위치를 무엇이라고 하는가?
4. 배열의 크기보다 더 큰 인덱스를 사용하면 어떻게 되는가?
5. 배열의 크기를 나타낼 때 변수를 사용할 수 있는가?



배열의 _____

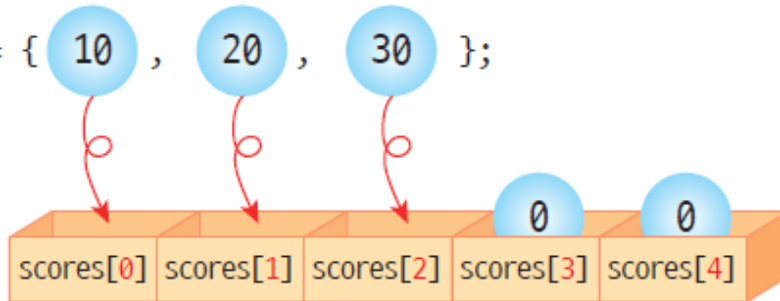


원소들의 초기값을 콤마로 분리하여 중괄호 안에 나열합니다.



배열의 초기화

```
int scores[5] = { 10 , 20 , 30 };
```

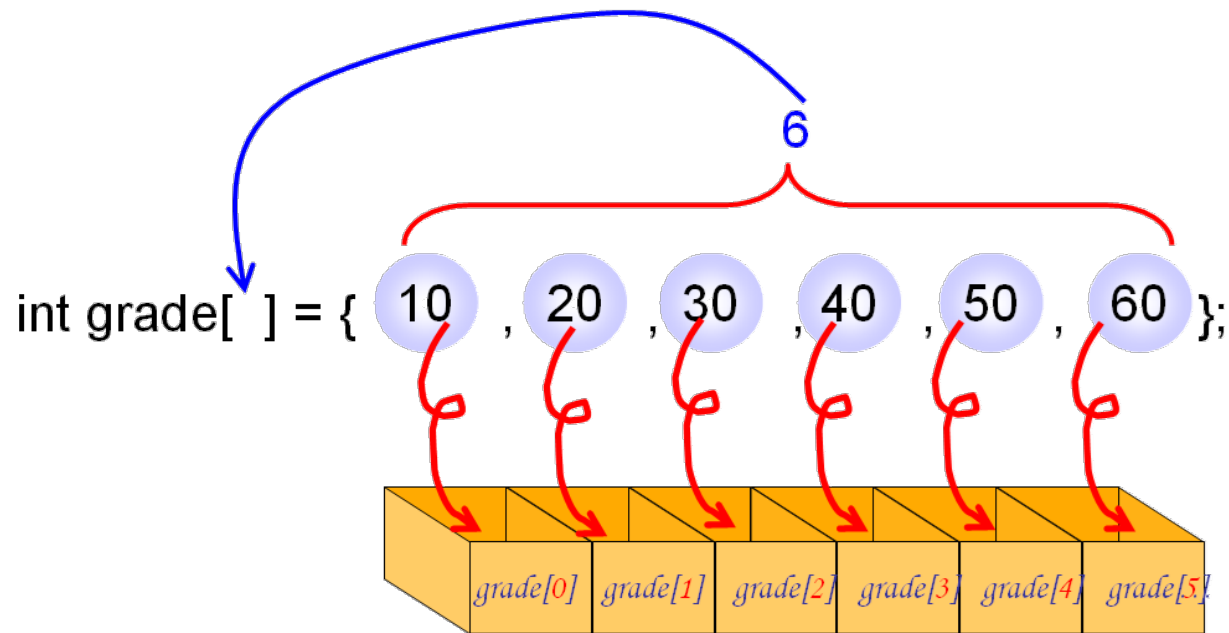


초기값을 일부만 주면 나머지
원소들은 0으로 초기화됩니다.



배열의 선언과 초기화

- 배열의 크기가 주어지지 않으면 자동적으로 초기값의 개수만큼이 배열의 크기로 잡힌다.



배열 초기화 예제

```
#include <stdio.h>
int main(void)
{
    int grade[5] = { 31, 63, 62, 87, 14 };
    int i;

    for(i = 0; i < 5; i++)
        printf("grade[%d] = %d\n", i, grade[i]);

    return 0;
}
```



```
grade[0] = 31
grade[1] = 63
grade[2] = 62
grade[3] = 87
grade[4] = 14
```

배열 초기화 예제 2

```
#include <stdio.h>
int main(void)
{
    int grade[5] = { 31, 63 };
    int i;

    for(i = 0; i < 5; i++)
        printf("grade[%d] = %d\n", i, grade[i]);

    return 0;
}
```



```
grade[0] = 31
grade[1] = 63
grade[2] = 0
grade[3] = 0
grade[4] = 0
```

배열 초기화 예제 3

```
#include <stdio.h>
int main(void)
{
    int grade[5] ;
    int i;

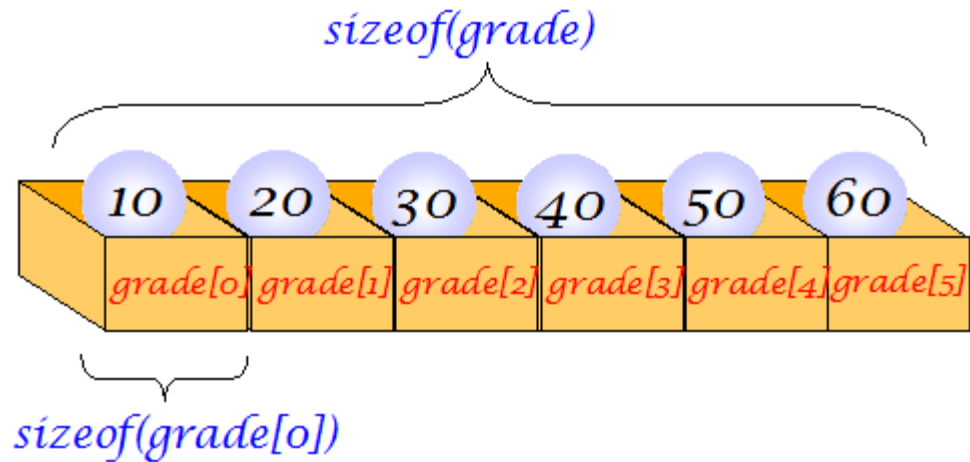
    for(i = 0; i < 5; i++)
        printf("grade[%d] = %d\n", i, grade[i]);

    return 0;
}
```



```
grade[0]=4206620
grade[1]=0
grade[2]=4206636
grade[3]=2018779649
grade[4]=1
```

배열 원소의 개수 계산



```
int grade[] = { 1, 2, 3, 4, 5, 6 };
```

```
int i, size;
```

```
size = sizeof(grade) / sizeof(grade[0]);
```

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

```
    printf("%d ", grade[i]);
```


배열의 복사

```
int grade[SIZE];  
int score[SIZE];
```

```
score = grade; // 컴파일 오류!
```

잘못된 방법



```
#include <stdio.h>  
#define SIZE 5
```

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

```
    int i;  
    int a[SIZE] = {1, 2, 3, 4, 5};  
    int b[SIZE];
```

```
    for(i = 0; i < SIZE; i++)  
        b[i] = a[i];
```

```
    return 0;
```

```
}
```

올바른 방법

배열의 비교



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

```
#define SIZE 5
```

```
int main(void)
```

```
{
```

```
    int i;
```

```
    int a[SIZE] = { 1, 2, 3, 4, 5 };
```

```
    int b[SIZE] = { 1, 2, 3, 4, 5 };
```

```
    if( a == b )
```

```
        printf("같습니다.\n");
```

```
    else
```

```
        printf("다릅니다.\n");
```

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

```
    {
```

```
        if ( a[i] != b[i] )
```

```
        {
```

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

```
            return 0;
```

```
        }
```

```
    }
```

```
    printf("같습니다.\n");
```

```
    return 0;
```

```
}
```

원소를 일일이
비교한다

중간 점검

1. 배열 `a[6]`의 원소를 1, 2, 3, 4, 5, 6으로 초기화하는 문장을 작성하라.
2. 배열의 초기화에서 초기값의 개수가 배열 원소의 개수보다 적은 경우에는 어떻게 되는가? 또 반대로 많은 경우에는 어떻게 되는가?
3. 배열의 크기를 주지 않고 초기값의 개수로 배열의 크기를 결정할 수 있는가?
4. 배열 `a`, `b`를 `if(a==b)`와 같이 비교할 수 있는가?
5. 배열 `a`에 배열 `b`를 `a=b;`와 같이 대입할 수 있는가?

예제) 주사위 던지기

- 주사위를 10000번 던져서 각면이 나오는 횟수를 출력하여 보자



- 주사위를 던지는 동작은 난수 발생기를 이용

```
int face;  
face = rand()%6; // 0, 1, 2, 3, 4, 5 의 값이 랜덤하게 face에 저장
```

예제) 주사위 던지기

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

#define SIZE 6

int main(void)
{
    int freq[SIZE] = { 0 };           // 주사위의 면의 빈도를 0으로 초기화한다.
    int i;

    for(i = 0; i < 10000; i++)        // 주사위를 10000번 던진다.
        ++freq[ rand() % 6 ];        // 해당면의 빈도를 하나 증가한다.

    printf("=====\n");
    printf("면   빈도\n");
    printf("=====\n");

    for(i = 0; i < SIZE; i++)
        printf("%3d   %3d \n", i, freq[i]);

    return 0;
}
```

예제) 극장 예약 시스템

- 배열을 이용하여 간단한 극장 예약 시스템을 작성
- 좌석은 10개
- 예약이 끝난 좌석은 1로, 예약이 안 된 좌석은 0으로 나타낸다.

좌석을 예약하시겠습니까?(y 또는 n) y

1 2 3 4 5 6 7 8 9 10

0 0 0 0 0 0 0 0 0 0

몇번째 좌석을 예약하시겠습니까?1

예약되었습니다.

좌석을 예약하시겠습니까?(y 또는 n) y

1 2 3 4 5 6 7 8 9 10

1 0 0 0 0 0 0 0 0 0

몇번째 좌석을 예약하시겠습니까?1

이미 예약된 자리입니다. 다른 좌석을 선택하세요

좌석을 예약하시겠습니까?(y 또는 n) n



예제) 극장 예약 시스템

- 알고리즘

```
while(1)
```

```
    사용자로부터 예약 여부(y 또는 n)를 입력받는다.
```

```
    if 입력 == 'y'
```

```
        현재의 좌석 배치표 seats[]를 출력한다.
```

```
        좌석 번호 i를 사용자로부터 입력받는다.
```

```
        if 좌석번호가 올바르면
```

```
            seats[i]=1
```

```
        else
```

```
            에러 메시지를 출력한다.
```

```
    else
```

```
        종료한다.
```

예제) 극장 예약 시스템

```
#include <stdio.h>
#define SIZE 10           // 배열의 크기는 기호상수로 정의하는 것이 편리
int main(void)
{
    char ans1;
    int ans2, i;
    int seats[SIZE] = {0}; // 예약 여부를 나타내는 배열을 선언하고 0으로 초기화
    while(1)
    {
        printf("좌석을 예약하시겠습니까?(y 또는n) ");
        scanf(" %c",&ans1); // 공백 문자는 제외하고 일반 문자만을 입력받는다.
```


예제) 극장 예약 시스템

```
if(ans1 == 'y')
{
    printf("-----\n");
    printf(" 1 2 3 4 5 6 7 8 9 10\n");
    printf("-----\n");
    for(i = 0; i < SIZE; i++)
        printf(" %d", seats[i]);
    printf("\n");
    printf("몇번째 좌석을 예약하시겠습니까?");
    scanf("%d",&ans2);
```

예제) 극장 예약 시스템

```
    if(ans2 <= 0 || ans2 > SIZE) {  
        printf("1부터 10사이의 숫자를 입력하세요\n");  
        continue;  
    }  
    if(seats[ans2-1] == 0) { // 예약되지 않았으면  
        seats[ans2-1] = 1;  
        printf("예약되었습니다.\n");  
    }  
    else // 이미 예약되었으면  
        printf("이미 예약된 자리입니다.\n");  
}  
else if(ans1 == 'n')  
    return 0;  
}  
return 0;  
}
```

배열 원소 역순 출력



```
#include <stdio.h>
#define SIZE 5

int main(void)
{
    int data[SIZE];
    int i;

    for(i = 0; i < SIZE; i++)    // 정수를 입력받는 루프
    {
        printf("정수를 입력하시오:");
        scanf("%d", &data[i]);
    }

    for(i = SIZE - 1; i >= 0; i--)    // 역순으로 출력하는 루프
    {
        printf("%d\ n", data[i]);
    }
    return 0;
}
```



정수를 입력하시오:10
정수를 입력하시오:20
정수를 입력하시오:30
정수를 입력하시오:40
정수를 입력하시오:50
50
40
30
20
10

예제



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

int main(void)
{
    int grade[STUDENTS] = { 30, 20, 10, 40, 50 };
    int i, s;

    for(i = 0; i < STUDENTS; i++)
    {
        printf("번호 %d: ", i);
        for(s = 0; s < grade[i]; s++)
            printf("*");
        printf("\n");
    }

    return 0;
}
```



```
번호 0: *****
번호 1: *****
번호 2: *****
번호 3: *****
번호 4: *****
```

예제) 최소값 탐색

- 우리는 인터넷에서 상품을 살 때, 가격 비교 사이트를 통하여 가장 싼 곳을 검색한다.



- 일반적으로 배열에 들어 있는 정수 중에서 **최소값**을 찾는 문제와 같다.






예제) 최소값 탐색

- 실행 결과

1 2 3 4 5 6 7 8 9 10

28 81 60 83 67 10 66 97 37 94

최소값은 10입니다.

Store	Certified rating	Inventory	Price	Total price
 Your Trusted Source since 1983	★★★★★ Rate this store See store profile	In stock Great Accessory Prices	Price: \$312.00 Tax: \$0.00 Shipping: Free	\$312.00 Your best price Shop now
	★★★★★ Rate this store See store profile	In stock	Price: \$312.95 Tax: \$0.00 Shipping: Free	\$312.95 Shop now
	★★★★☆ Rate this store See store profile	In stock	Price: \$312.95 Tax: \$0.00 Shipping: Free	\$312.95 Shop now
	★★★★☆ Rate this store See store profile	In stock	Price: \$313.00 Tax: \$0.00 Shipping: Free	\$313.00 Shop now
	Not yet rated Rate this store See store profile	In stock	Price: \$316.50 Tax: \$0.00 Shipping: Free	\$316.50 Shop now

예제) 최소값 탐색

- 알고리즘

배열 *prices[]*의 원소를 난수로 초기화한다.

일단 첫 번째 원소를 최소값 *minium*이라고 가정한다.

for(*i*=1; *i*<배열의 크기; *i*++)

if (*prices[i]* < *minimum*)

minimum = *prices[i]*

반복이 종료되면 *minimum*에 최소값이 저장된다.

예제) 최소값 탐색

```
#include <stdio.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define SIZE 10
int main(void)
{
    int prices[SIZE] = { 0 }; // 배열을 0으로 초기화
    int i, minimum;
    printf("-----\n");
    printf("1 2 3 4 5 6 7 8 9 10\n");
    printf("-----\n");
    srand( (unsigned)time( NULL ) );
    for(i = 0; i < SIZE; i++){
        prices[i] = (rand()%100)+1;
        printf("%-3d ", prices[i]);
    }
    printf("\n\n");
}
```

물건의 가격
출력

// 배열에 난수 입력
// %-3d는 3자리의 필드에 왼쪽
정렬하여 출력하라는 것을 의미


```
minimum = prices[0];
```

첫 번째 배열 원소를 최소
값으로 가정

```
for(i = 1; i < SIZE; i++)  
{
```

```
    if( prices[i] < minimum )  
        minimum = prices[i];
```

```
}
```

```
printf("최소값은 %d입니다.\n", minimum);
```

```
return 0;
```

```
}
```

현재의 최소값보다 배열 원소가 작으면, 배열 원소를 최소값으로 복사한다.



^ ^ ^ ^ V V V V V



실습 (1/2)

1. 주사위 던지기
2. 극장 예약 시스템
3. 극장 예약 시스템 프로그램에서는 한 명만 예약할 수 있다. 하지만 극장에 혼자서 가는 경우는 드물다. 따라서 한번에 2명을 예약할 수 있도록 2번 프로그램을 변경하여 보자.
4. 최소값 찾기
5. 최소값 찾기 프로그램에서는 최소값을 계산하였다. 이번에는 배열의 원소 중에서 최대값을 찾도록 변경하여 보자. 변수 이름도 적절하게 변경하라.

실습 (2/2)

- 03 2개의 정수 배열 a, b를 받아서 대응되는 배열 요소가 같은지를 검사하는 함수 `array_equal(int a[], int b[], int size)`를 작성하고 테스트하라. 이 함수는 `a[0]`와 `b[0]`, `a[1]`과 `b[1]`, ... , `a[size-1]`와 `b[size-1]`가 같은지를 검사한다. 만약 전체 요소가 같다면 1을 반환하고 그렇지 않으면 0을 반환한다.



```
C:\WINDOWS\system32\cmd.exe
1 2 3 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
2개의 배열은 다름
```

HINT 반복 루프를 이용해서 배열의 각 요소가 같은지를 검사한다. 만약 하나라도 다르면 0을 바로 반환하면 된다.

- 04 2개의 정수 배열 a, b를 받아서 배열 a의 요소를 배열 b로 복사하는 함수 `array_copy(int a[], int b[], int size)`를 작성하고 테스트하라. 이 함수는 `a[0]`를 `b[0]`에, `a[1]`를 `b[1]`에, ... , `a[size-1]`을 `b[size-1]`에 대입한다. 이 함수의 반환값은 없다.



```
C:\WINDOWS\system32\cmd.exe
1 2 3 0 0 0 0 0 0 0
1 2 3 0 0 0 0 0 0 0
```

HINT 반복 루프를 이용해서 배열의 각 요소를 복사한다.

숙제 (1/2)

- 1 배열 days[]를 아래와 같이 초기화하고 배열 요소의 값을 다음과 같이 출력하는 프로그램을 작성하라.
31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31



```
C:\WINDOWS\system32\cmd.exe
8월은 31일까지 있습니다.
9월은 30일까지 있습니다.
10월은 31일까지 있습니다.
11월은 30일까지 있습니다.
12월은 31일까지 있습니다.
```

HINT 배열을 초기화하려면 `int x[] = { 1, 2 };`와 같이 한다.

- 2 크기가 10인 1차원 배열에 난수를 저장한 후에, 최대값과 최소값을 출력하는 프로그램을 작성하라. 난수는 `rand()` 함수를 호출하여 생성하라.



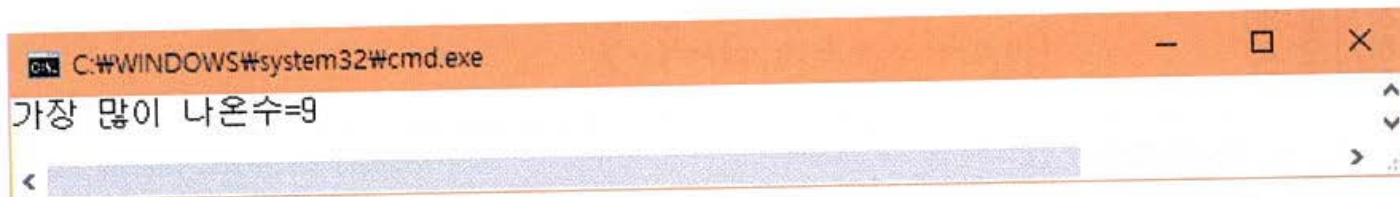
```
C:\WINDOWS\system32\cmd.exe
최대값은 29358
최소값은 41
```

HINT `x[i] = rand();` // 난수를 생성하여서 i번째 배열 요소에 대입한다.

숙제 (2/2)

- 3

0부터 9까지의 난수를 100번 생성하여 가장 많이 생성된 수를 출력하는 프로그램을 작성하라. 난수는 `rand()` 함수를 사용하여 생성하라.



HINT 본문의 빈도수 구하는 예제를 참고한다. 0에서 9까지의 난수는 `rand()%10`으로 구할 수 있다.

Q & A

