

배열 Array

2022

국민대학교 소프트웨어학부

학습 목표

- 배열에 대해 알아봅니다.
 - 배열과 원소, 원소를 가리키는 인덱스를 이해한다.
- GDB를 사용하여 배열을 추적해봅니다.

동일한 자료형 N개를 저장하려면?

- 실습으로 한번 확인해보기

실습 4-1

문자 변수를 10 개 만들어 'a'를 10개 저장하고
그 값을 다 출력하는 프로그램을 작성해 보자.

```
ejin@ejin-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejin@ejin-VirtualBox:~/C2020$ ./a10
a
a
a
a
a
a
a
a
a
a
a
ejin@ejin-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

세미콜론(;)이
한문장의
끝을 나타낸다.

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

int main(){
    char v1,v2,v3,v4,v5,v6,v7,v8,v9,v10;

    v1 = 'a';
    v2 = 'a';
    v3 = 'a';
    v4 = 'a';
    v5 = 'a'; v6 = 'a'; v7 = 'a'; v8 = 'a'; v9 = 'a';
    v10 = 'a';

    cout << v1 << endl;
    cout << v2 << endl; cout << v3 << endl;
    cout << v4 << endl << v5 << endl << v6 << endl;
    cout << v7 << endl << v8 << endl << v9 << endl;
    cout << v10 << endl;

    return 0;
}
```

한 줄에 여러 문장을
쓸 수 있습니다.

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

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

int main(){
    char v1,v2,v3,v4,v5,v6,v7,v8,v9,v10;

    v1 = 'a';
    v2 = 'a';
    v3 = 'a';
    v4 = 'a';
    v5 = 'a'; v6 = 'a'; v7 = 'a'; v8 = 'a'; v9 = 'a';
    v10 = 'a';

    cout << v1 << endl;
    cout << v2 << endl; cout << v3 << endl;
    cout << v4 << endl << v5 << endl << v6 << endl;
    cout << v7 << endl << v8 << endl << v9 << endl;
    cout << v10 << endl;

    return 0;
}
```

cout과 << 연산자
체인 형태로 사용할
수 있다.

endl?

개행문자

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

int main(){
    char v1,v2,v3,v4,v5,v6,v7,v8,v9,v10;

    v1 = 'a';
    v2 = 'a';
    v3 = 'a';
    v4 = 'a';
    v5 = 'a'; v6 = 'a'; v7 = 'a'; v8 = 'a'; v9 = 'a';
    v10 = 'a';

    cout << v1 << endl;
    cout << v2 << endl; cout << v3 << endl;
    cout << v4 << endl << v5 << endl << v6 << endl;
    cout << v7 << endl << v8 << endl << v9 << endl;
    cout << v10 << endl;

    return 0;
}
```

변수 v1, v2 ... v10을 배열로 변경하면
...

반복문 for 를 사용하여
프로그램을 변경할 수 있다.!

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

int main(){
    char v1,v2,v3,v4,v5,v6,v7,v8,v9,v10;

    v1 = 'a';
    v2 = 'a';
    v3 = 'a';
    v4 = 'a';
    v5 = 'a'; v6 = 'a'; v7 = 'a'; v8 = 'a'; v9 = 'a';
    v10 = 'a';

    cout << v1 << endl;
    cout << v2 << endl; cout << v3 << endl;
    cout << v4 << endl << v5 << endl << v6 << endl;
    cout << v7 << endl << v8 << endl << v9 << endl;
    cout << v10 << endl;

    return 0;
}
```



```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
#include <iostream>
using namespace std;

int main(){
    char arr[10];
    int i;

    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```


프로그램이 간결해진다.!

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

int main(){
    char v1,v2,v3,v4,v5,v6,v7,v8,v9,v10;

    v1 = 'a';
    v2 = 'a';
    v3 = 'a';
    v4 = 'a';
    v5 = 'a'; v6 = 'a'; v7 = 'a'; v8 = 'a'; v9 = 'a';
    v10 = 'a';

    cout << v1 << endl;
    cout << v2 << endl; cout << v3 << endl;
    cout << v4 << endl << v5 << endl << v6 << endl;
    cout << v7 << endl << v8 << endl << v9 << endl;
    cout << v10 << endl;

    return 0;
}
```



```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
#include <iostream>
using namespace std;

int main(){
    char arr[10];
    int i;

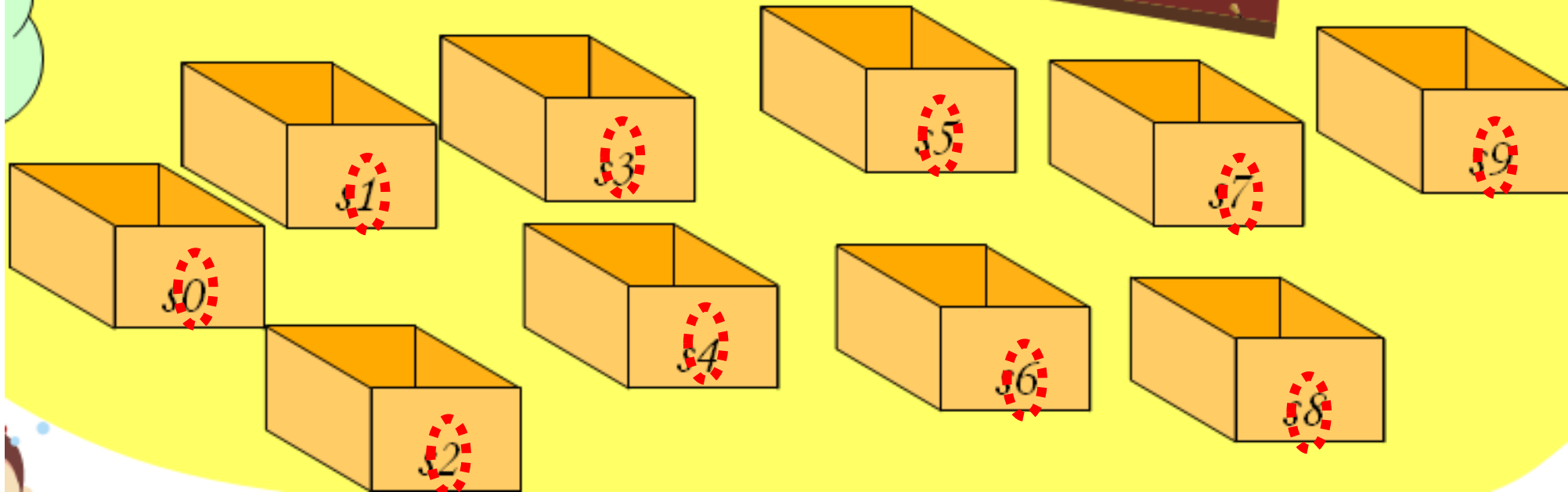
    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

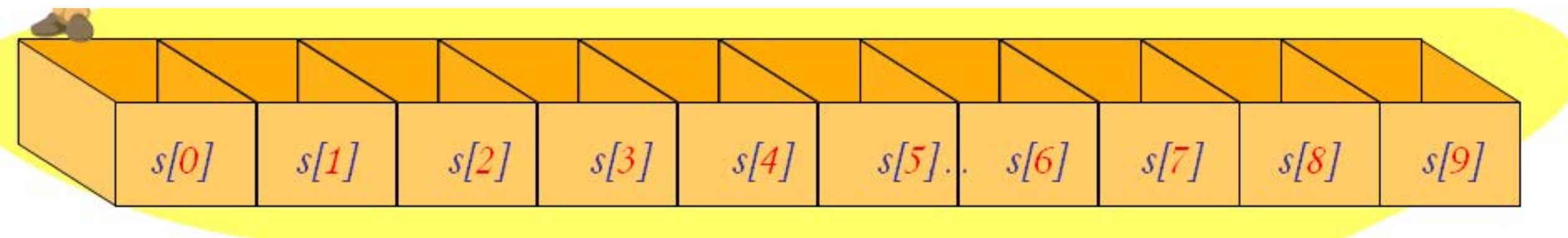
    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

배열의 크기가 100개라면?

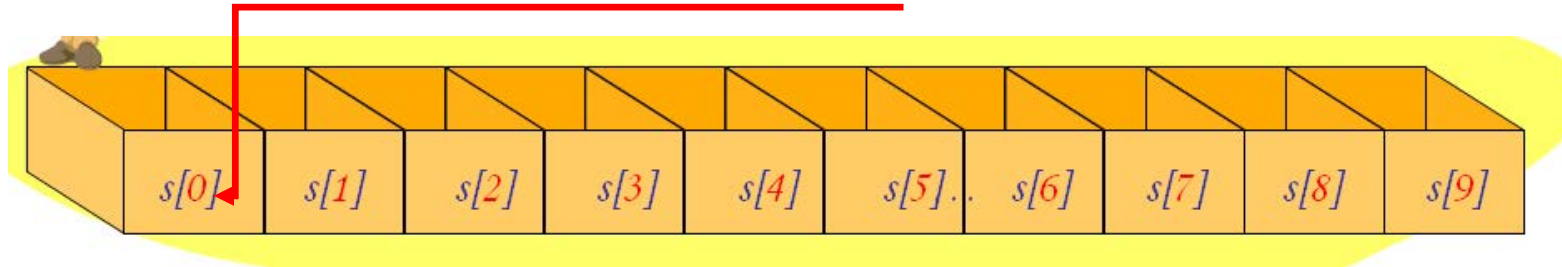
별도의 이름
을 가지니 조
작하기가 어
렵군!





배열이란?

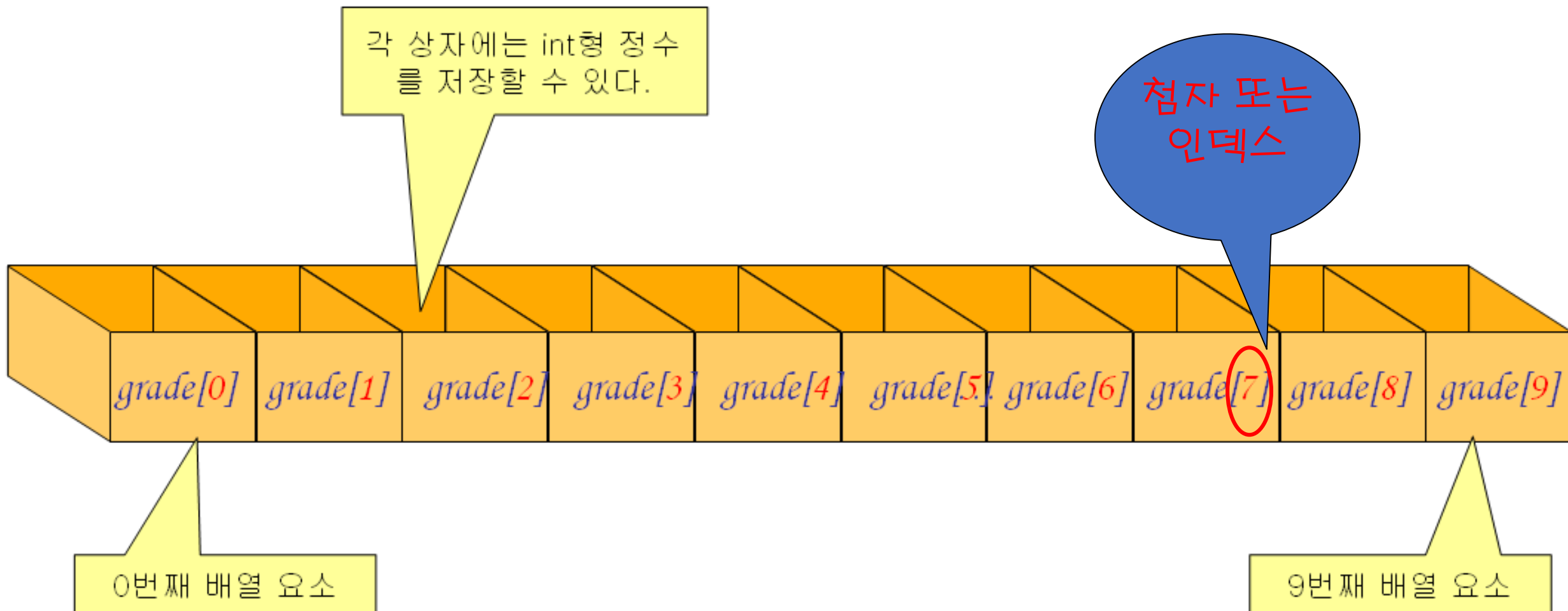
- **배열(array)**: 동일한 타입의 데이터가 여러 개 저장되어 있는 데이터 저장 장소
- 배열 안에 들어있는 각각의 데이터들은 정수로 되어 있는 번호(첨자=index)에 의하여 접근



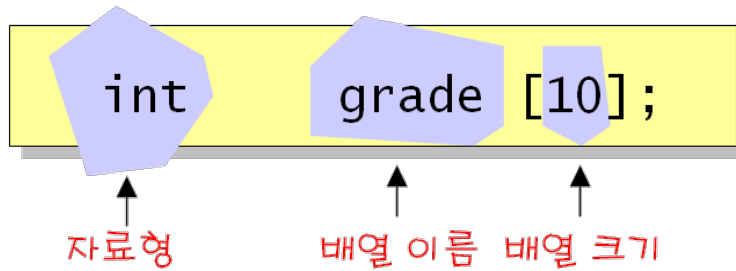
- 배열을 이용하면 여러 개의 값을 하나의 이름으로 처리할 수 있다.

배열 원소와 인덱스

- **인덱스(index):** 배열 원소의 번호



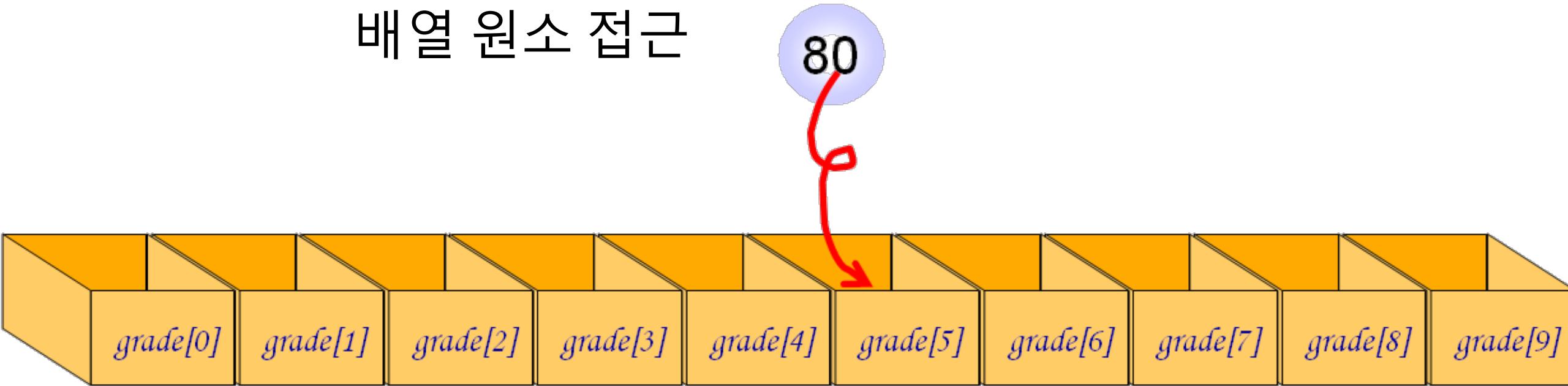
배열의 선언



- 자료형: 배열 원소들이 int형라는 것을 의미
- 배열 이름: 배열을 사용할 때 사용하는 이름이 grade
- 배열 크기: 배열 원소의 개수가 10개
- 인덱스(배열 번호)는 항상 0부터 시작한다.

```
int score[60];      // 60개의 int형 값을 가지는 배열 score
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`

인덱스

첨자

```
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
```

배열의 인덱스

- `int grade[10];`
- 위의 배열에서 사용할 수 있는 인덱스의 범위는 0에서 9까지 이다.

- 오류의 예

`cout << grade[10];` // 오류! 인덱스 10은 적합한 범위가 아니다.

`grade[10] = 99;` // 오류! 인덱스 10은 적합한 범위가 아니다.


```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
#include <iostream>
using namespace std;

int main(){
    char arr[10];
    int i;

    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
```

```
    char arr[10];
```

```
    int i;
```

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

```
        arr[i] = 'a';
```

```
    }
```

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

```
        cout << arr[i] << endl;
```

```
    return 0;
```

```
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
```

```
ejim@ejim-VirtualBox:~/C2020$
```

array size

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
```

```
    char arr[10];
```

```
    int i;
```

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

```
        arr[i] = 'a';
```

```
    }
```

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

```
        cout << arr[i] << endl;
```

```
    return 0;
```

```
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
```

```
ejim@ejim-VirtualBox:~/C2020$
```

순차적인
배열 원소
참조를 위한
첨자를 가리킬
변수 선언

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
    char arr[10];
    int i;
```

```
    for (i=0; i<10; i++){
        arr[i] = 'a';
    }
```

```
    for (i=0; i<10; i++)
        cout << arr[i] << endl;
```

```
    return 0;
```

```
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
```

```
ejim@ejim-VirtualBox:~/C2020$
```

반복문
for
미리보기

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
    char arr[10];
    int i;
```

```
    for (i=0; i<10; i++){
        arr[i] = 'a';
    }
```

```
    for (i=0; i<10; i++)
        cout << arr[i] << endl;
```

```
    return 0;
```

```
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
```

```
ejim@ejim-VirtualBox:~/C2020$
```

loop variable

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
    char arr[10];
    int i;


    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

loop body 의
이전에 수행



```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
    char arr[10];
    int i;

    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

loop body



```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
    char arr[10];
    int i;


    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

loop body 의
마지막에 수행




```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
    char arr[10];
    int i;

    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
i++;
i = i+1;
i += 1;
```

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
#include <iostream>
using namespace std;

int main(){
    char arr[10];
    int i;

    for (i=0; i<10; i++){
        arr[i] = 'a';
    }

    for (i=0; i<10; i++)
        cout << arr[i] << endl;

    return 0;
}
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
    char arr[10];
    int i;

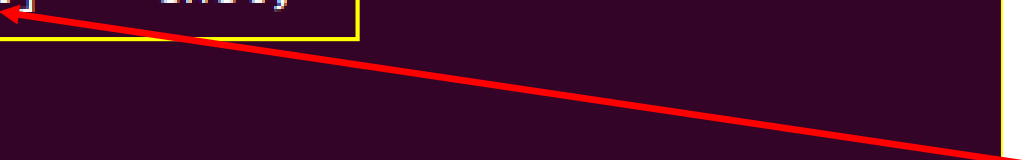
    for (i=0; i<10; i++){
        arr[i] = 'a';
    }
```

```
    for (i=0; i<10; i++)
        cout << arr[i] << endl;
```

```
    return 0;
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

array index
(0부터 시작)



```
ejim@ejim-VirtualBox:~/C2020$ cp a.cpp a10.cpp
ejim@ejim-VirtualBox:~/C2020$
```

```
ejim@ejim-VirtualBox:~/C2020$ cat a10.cpp
```

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

```
int main(){
```

```
    char arr[10];
```

```
    int i;
```

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

```
        arr[i] = 'a';
```

```
    }
```

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

```
        cout << arr[i] << endl;
```

```
    return 0;
```

```
}
```

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
```

```
ejim@ejim-VirtualBox:~/C2020$
```

array size

i++;
i = i+1;
i += 1;

array index (0부터 시작)

loop body 의 이전에 수행

loop variable

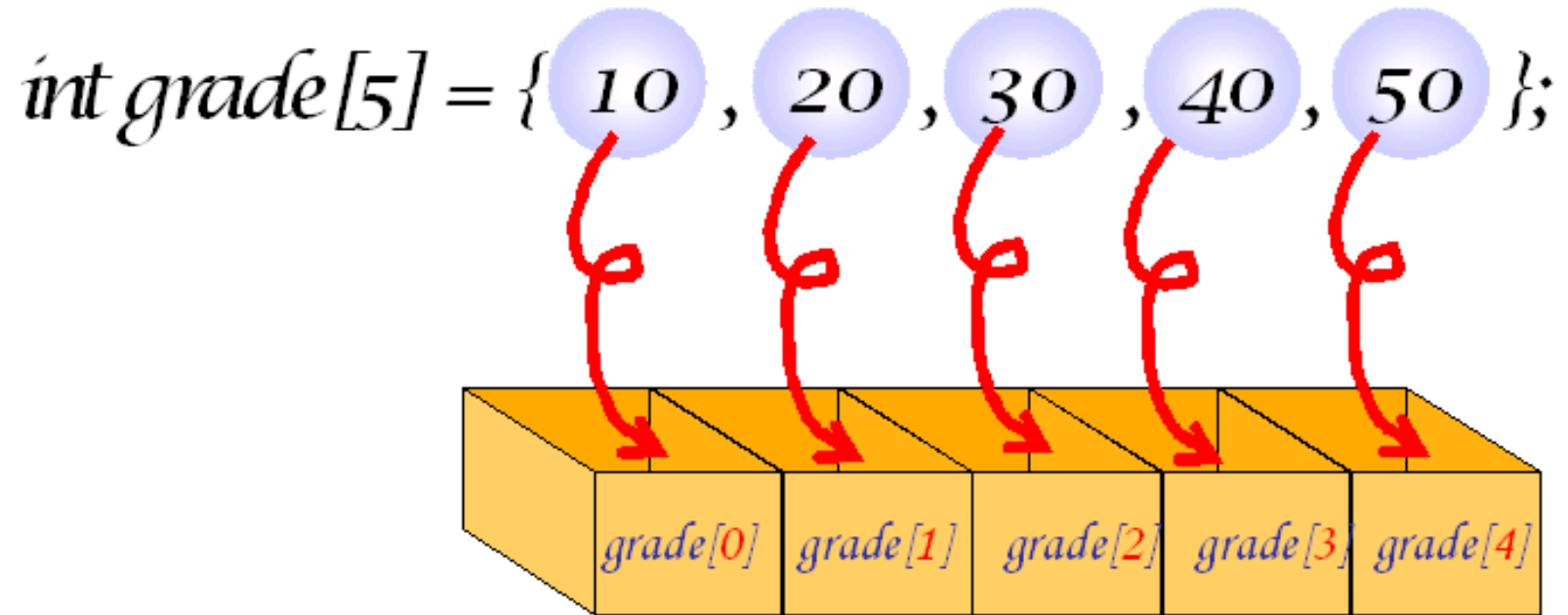
loop body

loop body 의 마지막에 수행

```
ejim@ejim-VirtualBox:~/C2020$ g++ -g -o a10 a10.cpp
ejim@ejim-VirtualBox:~/C2020$ ./a10
a
a
a
a
a
a
a
a
a
a
ejim@ejim-VirtualBox:~/C2020$
```

배열의 초기화

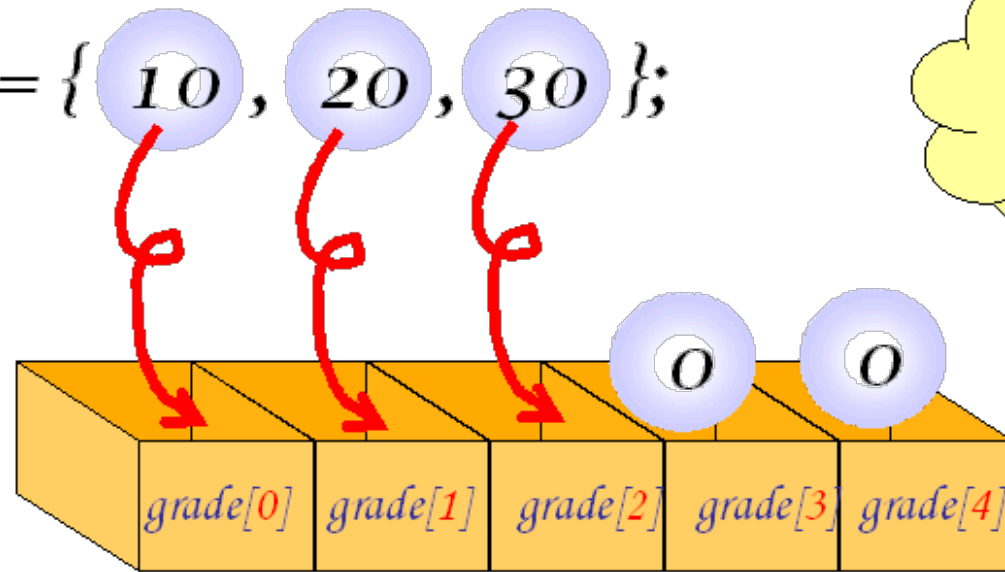
- `int grade[5] = { 10,20,30,40,50 };`



배열의 초기화

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

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

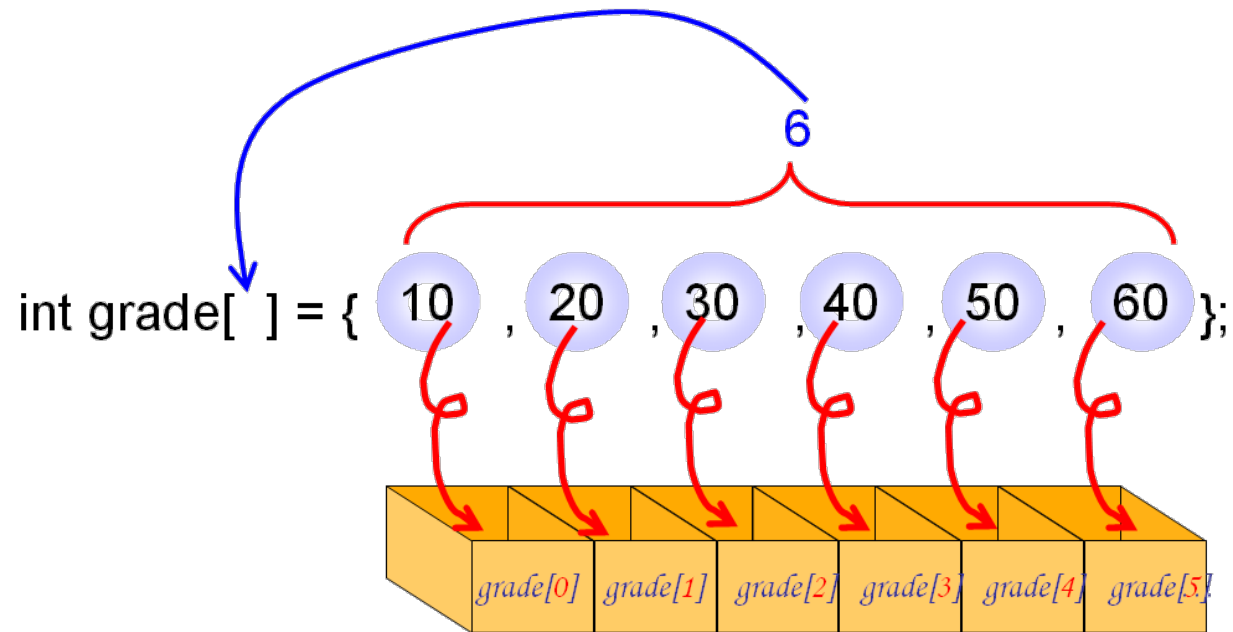


초기값을 일부만
주면 나머지 원소
들은 0으로 초기화
됩니다.



배열의 초기화

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



동적 배열

- C++에는 더 좋은 배열이 존재한다.
- STL 라이브러리로 제공



벡터(vector)

예제



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

int main()
{
    const int STUDENTS=5;
    int grade[STUDENTS] = { 30, 20, 10, 40, 50 };
    int i, s;

    for(i = 0; i < STUDENTS; i++)
    {
        cout << "번호 " << i;
        for(s = 0; s < grade[i]; s++)
            cout << "*";
        cout << endl;
    }

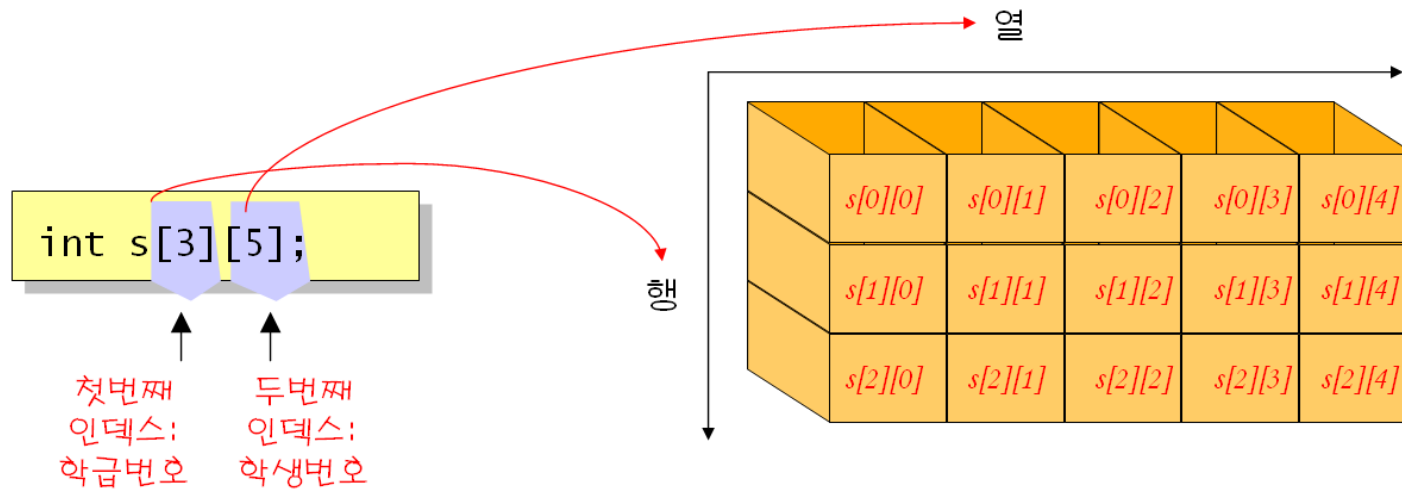
    return 0;
}
```



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

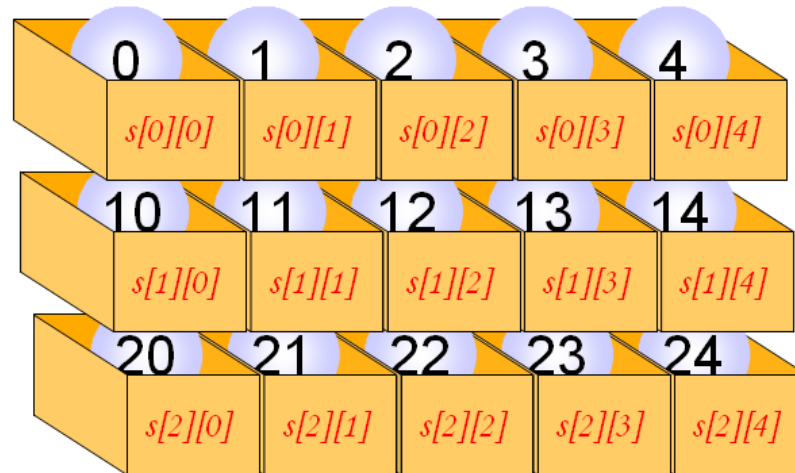
2차원 배열

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



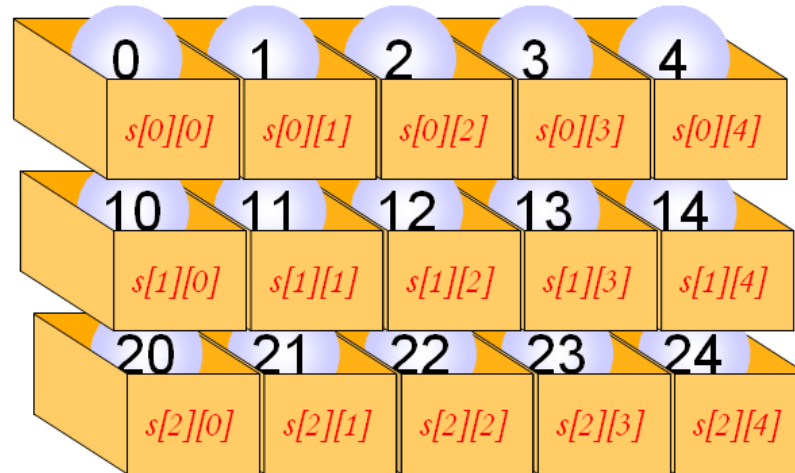
2차원 배열의 초기화

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



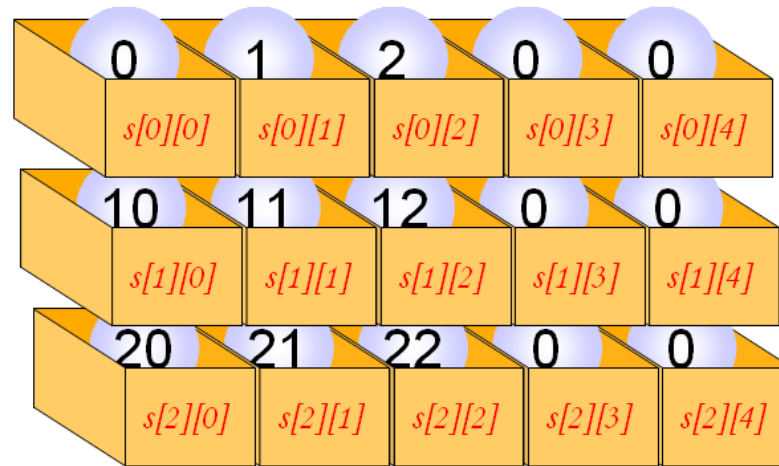
2차원 배열의 초기화

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



2차원 배열의 초기화

```
int s[ ][5] = {  
    { 0, 1, 2 },    // 첫 번째 행의 원소들의 초기값  
    { 10, 11, 12 }, // 두 번째 행의 원소들의 초기값  
    { 20, 21, 22 }  // 세 번째 행의 원소들의 초기값  
};
```



실습 4-2

- 2개의 행, 3개의 열을 갖는 2차원 정수 배열 arr를 선언하고,
10, -1, 3
2, 5, 6 의 값을 갖게 하라.
- 그 배열의 모든 원소의 값과 메모리 주소를 아래와 같이 출력하는 C++ 프로그램을 작성하여 컴파일하고 실행하라. (주소값은 각자 다를 것임)

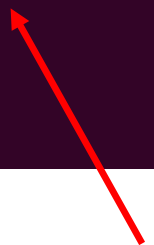
```
ejim@ejim-VirtualBox:~/C2020$ ./array1
arr[0][0] value: 10 address: 0x7ffd68216e60
arr[0][1] value: -1 address: 0x7ffd68216e64
arr[0][2] value: 3 address: 0x7ffd68216e68
arr[1][0] value: 2 address: 0x7ffd68216e6c
arr[1][1] value: 5 address: 0x7ffd68216e70
arr[1][2] value: 6 address: 0x7ffd68216e74
ejim@ejim-VirtualBox:~/C2020$
```

실습하기


```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int arr[2][3] = {{10, -1, 3},{2, 5, 6}};
6      int i,j;
7
8      for (i=0; i<2; i++)
9          for (j=0; j<3; j++) {
10             cout << "arr[" <<i << "][" << j << "] value: ";
11             cout << arr[i][j] << " address: " << &arr[i][j] << endl;
12         }
13     return 0;
14 }
```

```
cout << "arr[" << i << "][" << j << "] value: ";  
cout << arr[i][j] << " address: " << &arr[i][j] << endl;
```

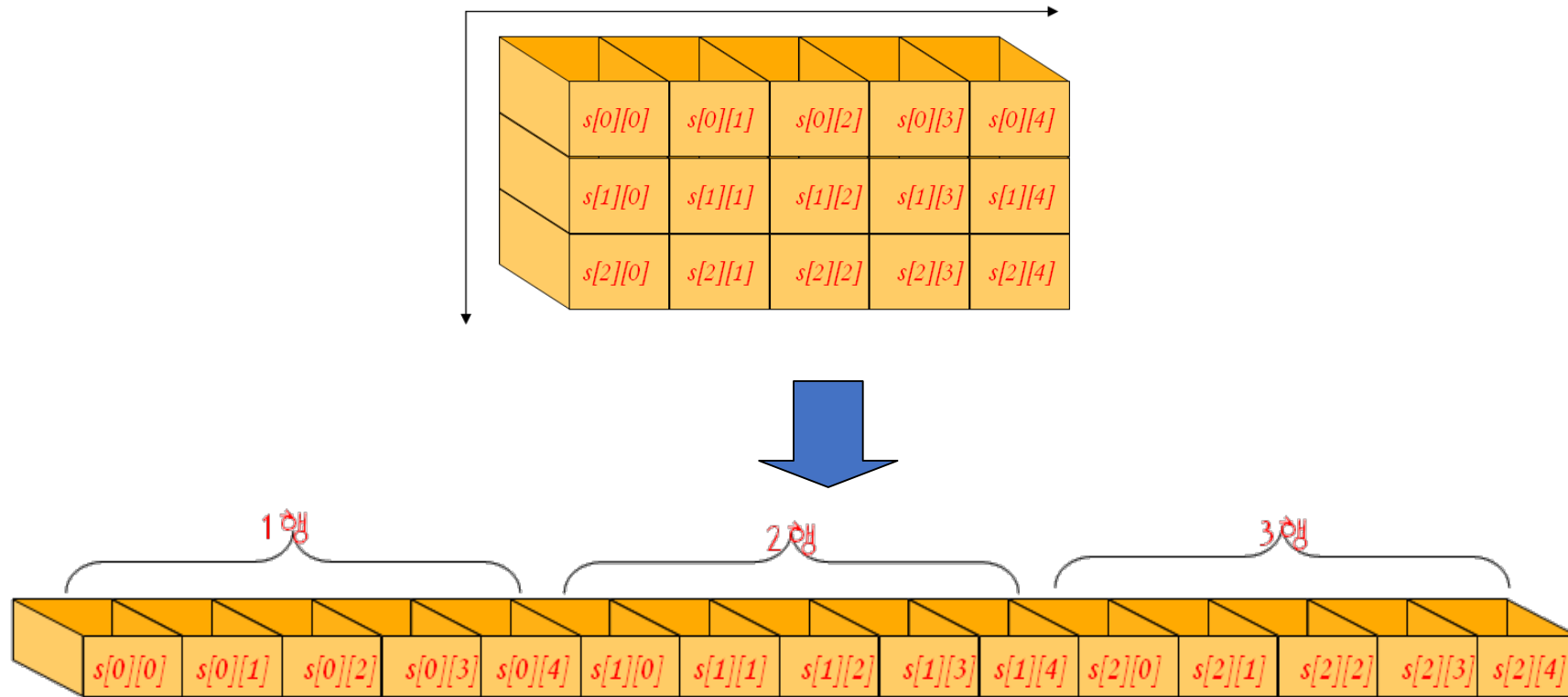
```
ejim@ejim-VirtualBox:~/C2020$ ./array1  
arr[0][0] value: 10 address: 0x7ffd68216e60  
arr[0][1] value: -1 address: 0x7ffd68216e64  
arr[0][2] value: 3 address: 0x7ffd68216e68  
arr[1][0] value: 2 address: 0x7ffd68216e6c  
arr[1][1] value: 5 address: 0x7ffd68216e70  
arr[1][2] value: 6 address: 0x7ffd68216e74  
ejim@ejim-VirtualBox:~/C2020$
```



왜 주소가 4만큼 차이가 날까?

2차원 배열의 저장

- 2차원 배열은 메모리 상에서 1차원적으로 저장된다.



row-major storage (행 우선 저장)

3차원 배열

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

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

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

int main()
{
    int s[6][3][5];    // 3차원 배열 선언
    int x, y, z;        // 3개의 인덱스 변수

    for(z=0;z<6;z++)
        for(y=0;y<3;y++)
            for(x=0;x<5;x++)
                s[z][y][x] = 10;

    return 0;
}
```

다차원 배열 예제



```
#include <iostream>
using namespace std;
const int CLASSES=3;
const int STUDENTS=5;
```



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

```
int main()
{
    int s[CLASSES][STUDENTS] = {
        { 0, 1, 2, 3, 4 },    // 첫번째 행의 원소들의 초기값
        { 10, 11, 12, 13, 14 }, // 두번째 행의 원소들의 초기값
        { 20, 21, 22, 23, 24 }, // 세번째 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];
        cout << "학급 " << clas << "의 평균 성적 = " << subtotal / STUDENTS << endl;
        total += subtotal;
    }
    cout << "전체 학생들의 평균 성적 = " << total / (CLASSES * STUDENTS) << endl;
    return 0;
}
```