1 1차원 배열

1 배열(Array)

같은 자료형을 가진 자료들을 나열하여 메모리에 연속으로 저장하여 만든 자료들의 그룹

- 요일을 나타내는 월요일, 화요일, 수요일, 목요일, 금요일, 토요일, 일요일을 각각 변수로 선언하면, 변수를 일곱 개 만들어 개별적으로 사용해야 함
- 하지만 하나로 묶어 배열로 만들면 배열을 한번만 선언해 만들 수 있고, 각 요일이 배열의 요소가 되어 다루기가 편해짐

1 배열(Array)

- 인덱스(Index)란?

  : 배열의 요소를 간단히 구별하기 위해 사용하는 번호

  - C에서 인덱스는 항상 0부터 시작함
  - 특정 배열요소를 사용할 경우에는 '배열이름[배열요소의 인덱스]'로 지정하고 변수처럼 사용하면 됨

1 배열(Array)

- 모든 자료형은 배열로 구성 가능
- 구성 형태에 따라 1차원 배열, 2차원 배열, 3차원 배열 등 다차원 배열로도 구성할 수 있음

2 1차원 배열 선언 형식

```
자료형 배열이름 [배열요소의 개수];
  ❶      ❷          ❸
```

❶ 배열의 자료형을 선언한다. 배열 요소는 모두 자료형이 같아야 하고, 배열 요소의 자료형이 배열의 자료형이 된다.

❷ 변수 이름과 같은 규칙으로 정한다.

❸ 대괄호([])를 사용해 배열 요소의 개수를 표시하는데, 배열 요소 개수가 배열 크기이다. 배열을 선언하면 메모리에 배열에 대한 공간이 할당되고 그 크기는 '자료형에 대한 메모리 할당 크기×배열 요소의 개수'이다.

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

〈변수이름 규칙〉

- 영문자, 숫자, 밑줄을 사용함
- 첫 글자는 숫자를 사용할 수 없음
- 알파벳 대문자와 소문자를 구분함
- 키워드(컴퓨터언어에서 미리 사용하기 위해 정의한 단어)나 예약어는 사용할 수 없음

2 1차원 배열 선언 형식

배열 선언 예	의미	배열 요소	메모리 할당 크기
char c[100];	char형 배열 요소 100개로 구성된 배열 c	c[0]~c[99]	1byte x 100
int i[100];	int형 배열 요소 100개로 구성된 배열 i	i[0]~i[99]	4byte x 100
short s[100];	short형 배열 요소 100개로 구성된 배열 s	s[0]~s[99]	2byte x 100
long l[100];	long형 배열 요소 100개로 구성된 배열 l	l[0]~l[99]	4byte x 100

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

2 1차원 배열 선언 형식

예제 1 자료형에 따른 메모리 할당 크기 확인하기

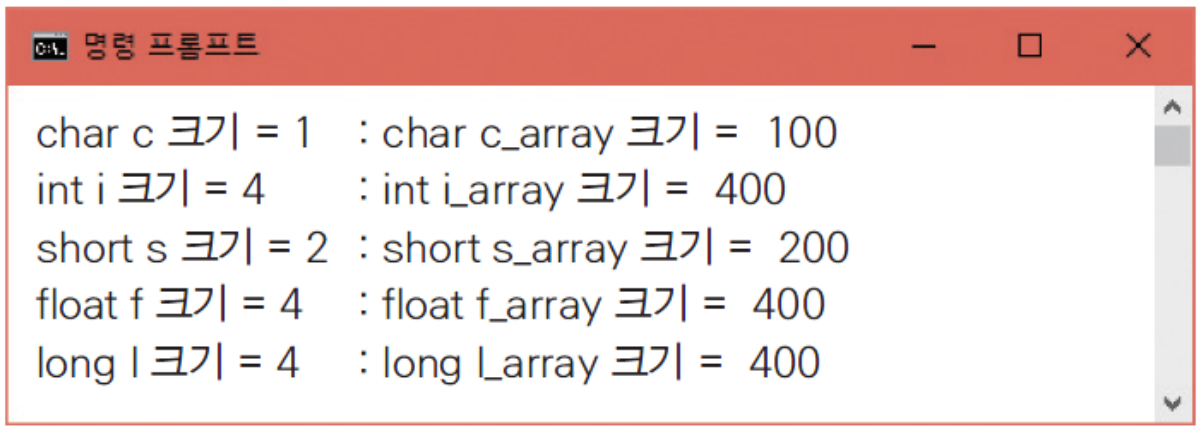
자료형에 대한 메모리 할당 크기는 컴퓨터에 따라(CPU 성능에 따라) 다를 수 있으므로 다음 예제를 통해 각 자료형에 대한 할당 크기를 확인할 수 있음

```
#include <stdio.h>
void main() {
  char c, c_array[100];          int i, i_array[100];
  short s, s_array[100];         float f, f_array[100];
  long l, l_array[100];
  printf("\n char c 크기 = %d \t: char c_array 크기 = %4d", sizeof(c), sizeof(c_array));
  printf("\n int i 크기 = %d \t: int i_array 크기 = %4d", sizeof(i), sizeof(i_array));
  printf("\n short s 크기 = %d \t: short s_array 크기 = %4d", sizeof(s), sizeof(s_array));
  printf("\n float f 크기 = %d \t: float f_array 크기 = %4d", sizeof(f), sizeof(f_array));
  printf("\n long l 크기 = %d \t: long l_array 크기 = %4d", sizeof(l), sizeof(l_array));
  getchar();  // 실행 창이 닫히지 않게 하기 위해 편의상 추가한 입력 대기 명령
}
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

2 1차원 배열 선언 형식

예제 1 자료형에 따른 메모리 할당 크기 확인하기



※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

2 1차원 배열 선언 형식

배열 선언과 메모리 할당 구조 예

- 40명의 중간고사 점수를 배열에 저장한다고 가정하면
  아래와 같이 배열을 선언하고 아래와 같은 구조로 메모리에 생성됨

```
int mid_score[40];
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 1차원 배열의 초기화

1차원 배열의 초기화 형식

- 초깃값 리스트에는 초깃값을 쉼표로 구분하여 나열하고, 초깃값들은 각 배열요소에 순서대로 지정됨

```
자료형 배열이름[배열크기] = { 초깃값 리스트 };
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 1차원 배열의 초기화

1차원 배열의 초기화 예 1

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

또는

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

배열의 모든 원소에 초깃값을 주면
배열 크기 생략 가능

또는

```
int A[5];
A[0] = 1;
A[1] = 2;
A[2] = 3;
A[3] = 4;
A[4] = 5;
```

(a) 1차원 배열의 초기화

	A[0]	A[1]	A[2]	A[3]	A[4]
A	1	2	3	4	5

(b) 메모리 할당 구조

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 1차원 배열의 초기화

- 1차원 배열의 초기화 예 2
  - 배열크기보다 초깃값을 적게 지정하면
    : 배열요소에 초깃값이 순서대로 할당되고
    나머지 배열요소는 주어진 초깃값이 없으므로
    기본값 0이 할당됨
  - 배열크기보다 초깃값을 크게 지정하면
    : 배열크기만큼의 초깃값만 배열요소에 할당되고
    나머지 초깃값은 메모리에는 할당되지만 배열
    영역의 밖에 저장도기 때문에 사용할 수 없는
    값이 됨

3 1차원 배열의 초기화

1차원 배열의 초기화 예 2

```
int A[5] = {1, 2, 3};
```

또는

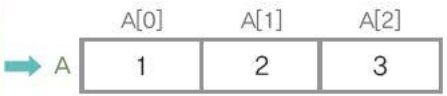
```
int A[5];
A[0] = 1;
A[1] = 2;
A[2] = 3;
```

(a) '초깃값의 개수 < 배열 크기'인 경우

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

또는

```
int A[3];
A[0] = 1;
A[1] = 2;
A[2] = 3;
A[3] = 4;
A[4] = 5;
```



(b) '초깃값의 개수 > 배열 크기'인 경우

※ 출처 : C로 배우는 쉬운 자료구조 (개정3판), 이지영, 한빛미디어

3 1차원 배열의 초기화

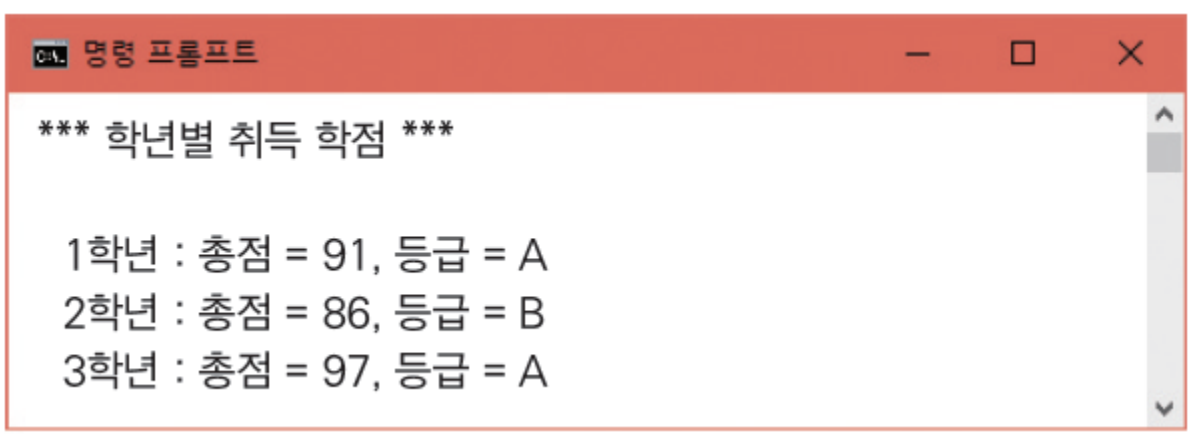
예제 2 학년별 취득 학점 입출력하기

```
void main() {
  int i;
  int score[3] = { 91, 86, 97 };    // 1차원 배열 초기화
  char grade[3] = { 'A', 'B', 'A' };

  printf("\n *** 학년별 취득 학점 *** \n\n");
  for (i = 0; i<3; i++) {
        printf("%3d학년 : 총점 = %d, 등급 = %c\n", i + 1, score[i],
grade[i]);
  }
   getchar();
}
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 1차원 배열의 초기화

예제 2 학년별 취득 학점 입출력하기



※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 1차원 배열의 초기화

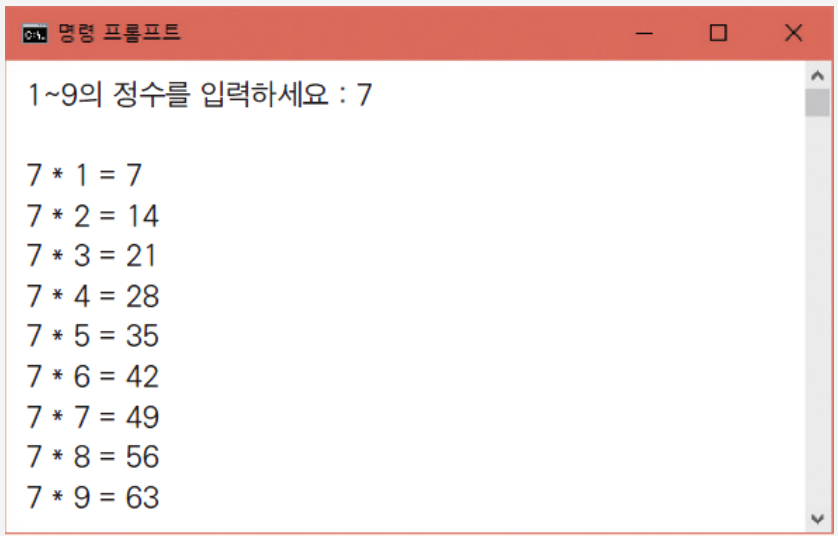
예제 3 입력한 숫자를 구구단으로 출력하기

```
void main() {
  int i = 0, n;    int multiply[9];
  printf("\n1~9의 정수를 입력하세요 : ");
  while (1) {
      scanf("%d", &n);
      if (n < 0 || n>9) printf("\n1~9의 정수를 입력하세요 : ");
      else  break;
  }
  printf("\n");
  for (i = 0; i < 9; i++) {
       multiply[i] = n*(i + 1);
       printf(" %d * %d = %d \n", n, (i + 1), multiply[i]);
  }
   getchar();    getchar();
}
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 1차원 배열의 초기화

예제 3 입력한 숫자를 구구단으로 출력하기



※ 출처 : C로 배우는 쉬운 자료구조 (개정3판), 이지영, 한빛미디어

2 문자 배열

1 문자 배열

- 문자의 나열한 것을 의미함
- “와 ”사이에 표시
- 문자열을 저장하기 위해서는 문자열을 구성하는 문자들을 연속적으로 저장해야 하기 때문에 char형 배열을 사용함
- 배열의 자료형은 문자 자료형(char)
- 문자 배열의 초기화는 문자열 그대로 지정하거나 초깃값 문자 리스트 사용한다.

1 문자 배열

- 문자배열을 문자열 “String” 으로 초기화하는 예
  - 문자열이 저장될 때는 문자열의 끝을 나타내는 ‘₩0’(null, 널문자)이 마지막에 추가되기 때문에 문자열을 저장할 메모리 크기는 실제 문자열 크기보다 1바이트가 더 커야 함

1 문자 배열

문자배열을 문자열 “String” 으로 초기화하는 예

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

```
char s1[10] = "String";
```

	s1[0]	s1[1]	s1[2]	s1[3]	s1[4]	s1[5]	s1[6]	s1[7]	s1[8]	s1[9]
s1	S	t	r	i	n	g	\0			

(a) 문자열을 사용한 초기화

```
char s2[10] = { 'S', 't', 'r', 'i', 'n', 'g' };
```

	s2[0]	s2[1]	s2[2]	s2[3]	s2[4]	s2[5]	s2[6]	s2[7]	s2[8]	s2[9]
s2	S	t	r	i	n	g				

(b) 초깃값 문자 리스트를 사용한 초기화

1 문자 배열

- s1을 초기값 문자 리스트를 사용 초기화

```
char s1[] = "String";
```

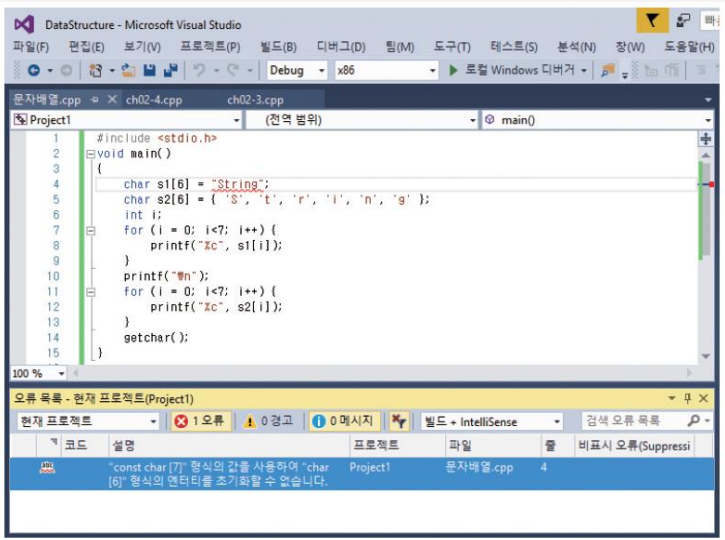
➡

```
char s1[] = { 'S', 't', 'r', 'i', 'n', 'g', '\0' };
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

1 문자 배열

- 배열 s1과 배열 s2 크기를 똑같이 6으로 선언 시 저장 공간 부족 오류



※ 출처 : C로 배우는 쉬운 자료구조 (개정3판), 이지영, 한빛미디어

- s1은 마지막에 '₩0'을 저장할 공간이 부족하므로 컴파일 오류가 남

1 문자 배열

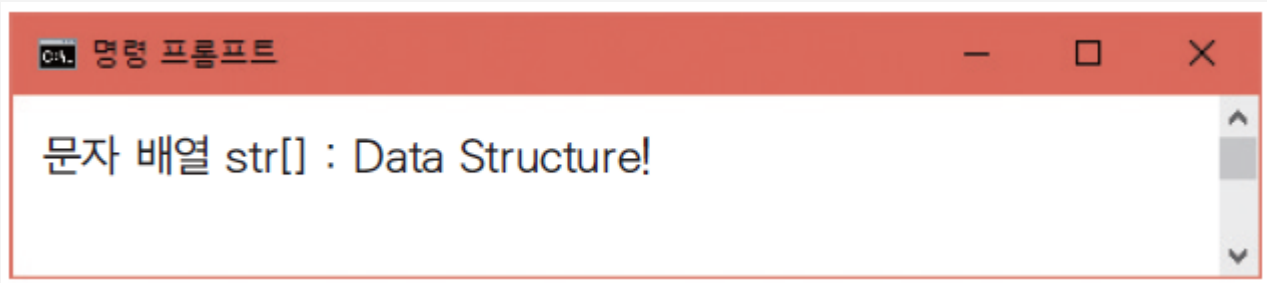
예제 1 문자 배열에 문자열을 저장하고 출력하기

```
void main() {
   char str[20] = "Data Structure!";
   int i;

   printf("\n문자 배열 str[] : ");

   for (i = 0; str[i]; i++) printf("%c", str[i]);

   getchar();
}
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

1 문자 배열

예제 1 문자 배열에 문자열을 저장하고 출력하기



※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

1 문자 배열

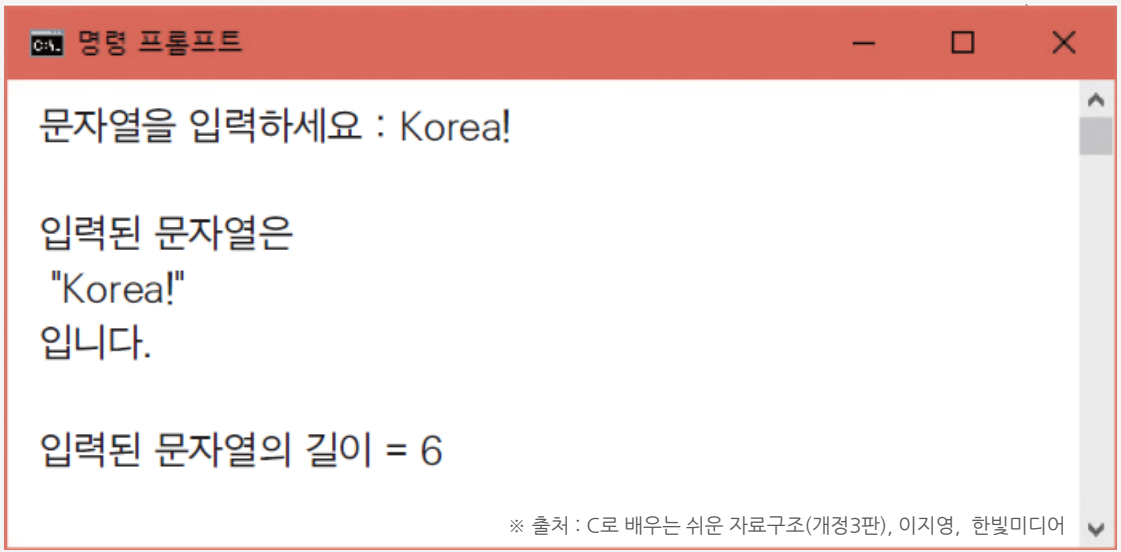
예제 2 입력한 문자열의 길이 계산하기

```
void main() {
   int i, length = 0;    char str[50];
   printf("\n문자열을 입력하세요 : ");
   gets(str);  // gets_s(str);
   printf("\n입력된 문자열은 \n \"");
   for (i = 0; str[i]; i++) {
        printf("%c", str[i]);
        length += 1;
   }
   printf("\" \n입니다.");
   printf("\n\n입력된 문자열의 길이 = %d \n", length);getchar();
}
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

1 문자 배열

예제 2 입력한 문자열의 길이 계산하기



3 다차원 배열

3 | 다차원 배열

1 다차원 배열이란?

2차원 이상의 배열

- 다차원 배열의 선언
  - 배열의 차수 만큼 [배열크기] 항목을 추가

2 2차원 배열의 선언 형식

- 2차원 배열의 선언 형식은 배열의 배열이므로 선언은 1차원 배열과 같은 형식
- 늘어난 차수만큼 대괄호([ ])를 추가하고 그 안에 배열 크기를 지정함
- 2차원 배열은 차수가 2이므로 배열의 배열을 의미

2 2차원 배열의 선언 형식

자료형 배열이름 [배열크기] [배열크기]

- [배열크기] (행 개수) → 행 번호
- [배열크기] (열 개수) → 열 번호

} 논리적 구조에서의 표현

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

2 2차원 배열의 선언 형식

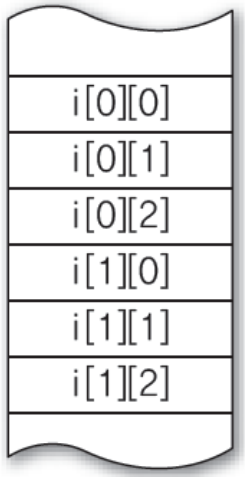
▶ 2차원 배열 선언의 논리적 구조와 물리적 구조 예

```
int i[2][3];
```

(a) 배열 선언

	열 번호 0	열 번호 1	열 번호 2
행 번호 0	i[0][0]	i[0][1]	i[0][2]
행 번호 1	i[1][0]	i[1][1]	i[1][2]

(b) 논리적 구조



(c) 물리적 구조

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 3차원 배열의 선언 형식

- 2차원 배열 선언 형식에서 차수만큼 대괄호([ ])를 추가하고 그 안에 배열 크기를 지정함
- 3차원 배열은 차수가 3이므로 배열의 배열을 의미

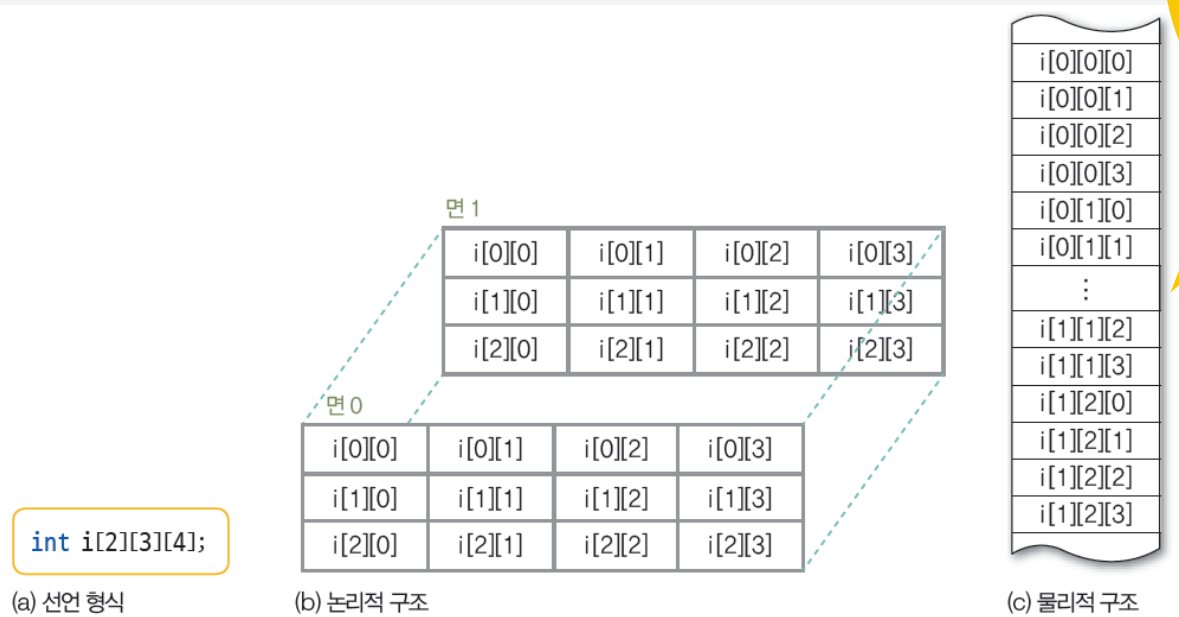
자료형 배열이름 [배열크기] [배열크기] [배열크기]

면 개수 행 개수 열 개수 → 논리적 구조에서의 표현

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

3 3차원 배열의 선언 형식

3차원 배열 선언 예



i는 면 2개와 행 3개와 열 4개로 구성된 **3차원 배열**

※ 출처 : C로 배우는 쉬운 자료구조 (개정3판), 이지영, 한빛미디어

4 다차원 배열의 초기화

- 초기값의 지정형태는 다차원 배열이 배열의 배열이라는 것을 생각하여 초기값을 구분하여 지정하거나, 1차원 배열처럼 초기값 리스트를 지정하여 순서대로 배열 요소의 초기값으로 설정

4 다차원 배열의 초기화

- 2차원 배열의 초기화와 논리적 구조 &
  행 크기를 생략한 2차원 배열의 초기화

1차원 배열과 마찬가지로 다음과 같이 배열 크기를 생략하고 초깃값을 지정할 수 있는데, 첫 번째 배열의 크기만 생략 할 수 있음

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

	[0]	[1]	[2]
[0]	1	2	3
[1]	4	5	6

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

※ 출처 : C로 배우는 쉬운 자료구조(개정 3판), 이지영, 한빛미디어

4 다차원 배열의 초기화

3차원 배열의 초기화와 논리적 구조

```
int i[2][3][4] =  {{1, 2, 3, 4 }, {5, 6, 7, 8}, {9, 10, 11, 12},
                   {13, 14, 15, 16}, {17, 18, 19, 20}, {21, 22, 23, 24}};
```

※ 출처 : C로 배우는 쉬운 자료구조(개정 3판), 이지영, 한빛미디어

4 다차원 배열의 초기화

예제 1 3차원 배열 입출력하기

```
void main() {
  int array[2][3][4];
  int i, j, k, value = 1;
  for (i = 0; i<2; i++) {
      for (j = 0; j<3; j++) {
          for (k = 0; k<4; k++) {
              array[i][j][k] = value;
              printf("\n array[%d][%d][%d] = %d", i, j, k, array[i][j][k]);
              value++;
          }
      }
  }
  getchar();
}
```

※ 출처 : C로 배우는 쉬운 자료구조(개정 3판), 이지영, 한빛미디어

명령 프롬프트

```
array[0][0][0] = 1
array[0][0][1] = 2
array[0][0][2] = 3
array[0][0][3] = 4
array[0][1][0] = 5
array[0][1][1] = 6
array[0][1][2] = 7
array[0][1][3] = 8
array[0][2][0] = 9
array[0][2][1] = 10
array[0][2][2] = 11
array[0][2][3] = 12
array[1][0][0] = 13
array[1][0][1] = 14
array[1][0][2] = 15
array[1][0][3] = 16
array[1][1][0] = 17
array[1][1][1] = 18
array[1][1][2] = 19
array[1][1][3] = 20
array[1][2][0] = 21
array[1][2][1] = 22
array[1][2][2] = 23
array[1][2][3] = 24
```

4 다차원 배열의 초기화

예제 1 3차원 배열 입출력하기

명령 프롬프트

```
array[0][0][0] = 1
array[0][0][1] = 2
array[0][0][2] = 3
array[0][0][3] = 4
array[0][1][0] = 5
array[0][1][1] = 6
array[0][1][2] = 7
array[0][1][3] = 8
array[0][2][0] = 9
array[0][2][1] = 10
array[0][2][2] = 11
array[0][2][3] = 12
array[1][0][0] = 13
array[1][0][1] = 14
array[1][0][2] = 15
array[1][0][3] = 16
array[1][1][0] = 17
array[1][1][1] = 18
array[1][1][2] = 19
array[1][1][3] = 20
array[1][2][0] = 21
array[1][2][1] = 22
array[1][2][2] = 23
array[1][2][3] = 24
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

5 문자 다차원 배열

- 문자배열 역시 1차원 배열을 묶어서 다차원 배열로 구성할 수 있음

5 문자 다차원 배열

- 2차원 문자 배열의 선언 예

```
char c[3][20];
```

문자 20개까지 저장할 수 있는 1차원 문자 배열 3개를 묶어서 구성한 2차원 문자 배열이 됨

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

5 문자 다차원 배열

- 2차원 문자 배열의 문자열 저장과 논리적 구조

(a)

```
char c[3][20]={ "Hong Gil Dong",
                "Computer Department",
                "Seoul Korea"};
```

(b)

```
strcpy(c[0], "Hong Gil Dong");
strcpy(c[1], "Computer Department");
strcpy(c[2], "Seoul Korea");
```

(c)

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]
[0]	H	o	n	g		G	i	l		D	o	n	g	\0						
[1]	C	o	m	p	u	t	e	r		D	e	p	a	r	t	m	e	n	t	\0
[2]	S	e	o	u	l		K	o	r	e	a	\0								

(a)는 선언과 동시에 초기화,
(b)는 문자열 복사함수인
strcpy()를 사용한 문자열 저장

※ 출처 : C로 배우는 쉬운 자료구조 (개정3판), 이지영, 한빛미디어

5 문자 다차원 배열

예제 2 3차원 배열을 이용해 문자 배열 입출력하기

```
void main() {
  int i, j, k;
  char student[2][3][20];
  for (i = 0; i<2; i++) {
    printf("\n 학생 %d의 이름 : ", i + 1);
    gets(student[i][0]);          // gets_s(student[i][0]);
    printf(" 학생 %d의 학과 : ", i + 1);
    gets(student[i][1]);          // get_s(student[i][1]);
    printf(" 학생 %d의 학번 : ", i + 1);
    gets(student[i][2]);          // get_s(student[i][2]);
  }
  for (i = 0; i<2; i++) {
     printf("\n\n 학생%d", i + 1);
     for (j = 0; j<3; j++) {
          printf("\n\t");
          for (k = 0; student[i][j][k] != '\0'; k++)
               printf("%c", student[i][j][k]);
     }
  }
  getchar();
}
```

※ 출처 : C로 배우는 쉬운 자료구조(개정3판), 이지영, 한빛미디어

명령 프롬프트

```
학생 1의 이름 : Hong Gil Dong
학생 1의 학과 : Computer Electronic
학생 1의 학번 : 201600101

학생 2의 이름 : Hong Gil Soon
학생 2의 학과 : Computer Science
학생 2의 학번 : 201600201


학생 1
        Hong Gil Dong
        Computer Electronic
        201600101

학생 2
        Hong Gil Soon
        Computer Science
        201600201
```