

학번: \_\_\_\_\_ 성명: \_\_\_\_\_.

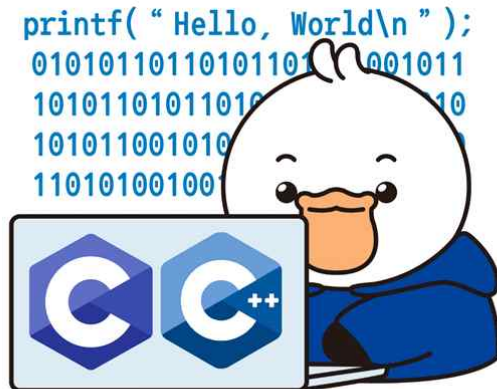
- 본 강의자료는 과학기술정보통신부 및 정보통신기획평가원에서 지원하는 『소프트웨어중심대학』 사업의 결과물입니다.
- 본 강의자료는 내용은 전재할 수 없으며, 인용할 때에는 반드시 과학기술정보통신부와 정보통신기획평가원의 '소프트웨어중심대학'의 결과물이라는 출처를 밝혀야 합니다.



## Part 6. 배열

## 목차

- 6.1 배열의 개요
- 6.2 1차원 배열
- 6.3 2차원 배열
- 6.4 다차원 배열
- 6.5 Q&A
- 6.6 실습 및 과제
- 6.7 참고문헌



### 6.1 배열의 개요

- 데이터 값 저장 및 활용
  - ✓ 학생들의 이름, 성적 등을 출력하는 프로그램 작성
  - ✓ 변수를 사용하여 아래 표의 정보 저장
    - name1, name2, ..., math3
    - 총 16개의 변수 필요
- 하나의 변수처럼 사용하면서 각각의 값을 저장 및 사용
  - ✓ 배열(Array)

이름	국어	영어	수학
박준홍	90	85	75
한윤성	50	50	60
이동욱	100	95	85
김수정	100	100	100

## 6.1 배열의 개요

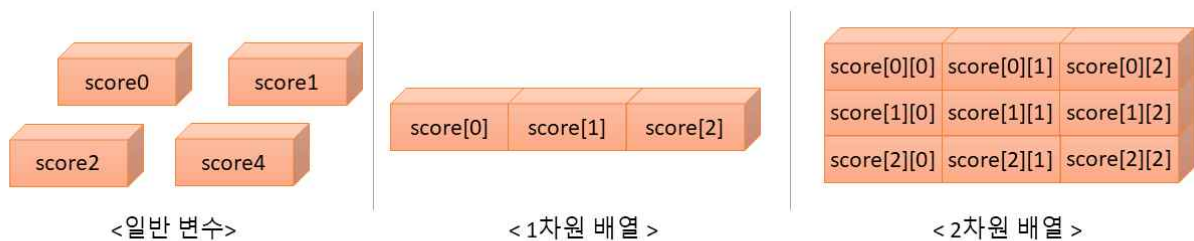
### ■ 배열(Array) 이란?

- ✓ 다수의 동일한 타입의 데이터를 저장하기 위해 연속적인 메모리의 공간을 할당
- ✓ 공간의 위치에 따라 번호를 부여하여 접근할 수 있도록 하는 데이터 구조
- ✓ 각각의 변수들을 원소(element, 요소)로 명칭
- ✓ 각 원소는 인덱스(index, 첨자)를 이용하여 접근
  - 인덱스 번호는 0부터 N-1 까지 각 원소에 할당
  - 반복문의 반복 제어 변수 등을 이용하면 손쉽게 제어 가능

## 6.1 배열의 개요

### ■ 배열 정의

- ✓ 배열의 3 요소
  - 배열 이름, 데이터 타입, 배열 크기
- ✓ 배열의 종류
  - 1차원 배열, 2차원 배열, 3차원 배열, 등



## 6.2 1차원 배열

### ■ 1차원 배열

#### ✓ 배열

➤ 인덱스라는 번호가 부여된 변수

#### ✓ 선언 방법

➤ 변수를 선언할 수 있는 데이터 타입이라면 배열로 선언 가능

• ex) int, char, double 등

배열 선언	데이터타입 배열이름[배열크기];
배열 선언 예	<pre>int score[30]; double grade[10]; char name[15];</pre>

## 6.2 1차원 배열

### ■ 배열의 구성

#### ✓ 데이터 타입

➤ 변수로 선언 가능한 모든 데이터 타입

#### ✓ 이름 선언 규칙

➤ 식별자 생성 규칙을 준수 시 사용 가능

#### ✓ 배열 크기 선언 방식

1. 대괄호([]) 안에 리터럴 상수 또는 매크로 상수로 크기 지정
2. 가변 크기 배열(VLA: Variable Length Arrays)를 사용하여 배열 크기 지정
  - c11 이후, MSVC 등 일부 컴파일러에서는 지원 하지 않음

## 6.2 1차원 배열

### 배열원소 접근

✓ int score[30]

- score[0]와 score[1], score[2] 등은 int형 변수와 동일하게 사용
- 각 배열 원소 접근 방법
  - 변수명[인덱스 번호]
- 인덱스 범위
  - 0 ~ (배열크기 - 1) → int score[30]의 인덱스 범위 : 0 ~ 29

## 6.2 1차원 배열

### 배열원소 주소 할당

✓ int arr[5]

- int : 4바이트 → 4바이트의 공간 5개를 연속적으로 할당 받아 총 20바이트의 공간을 사용
- 배열의 각 원소 위치
  - 배열의 시작 주소 + (인덱스 \* 원소의 크기)

주소	0000	0004	0008	0012	0016
원소	arr[0]	arr[1]	arr[2]	arr[3]	arr[4]
크기	4	4	4	4	4

## 6.2 1차원 배열

소스 6-1: 크기가 5인 int 형 배열 사용하기

```

1 #include<stdio.h>
2 int main(void)
3 {
4     int arr[5];
5     int i;
6     arr[0] = 22;
7     arr[1] = 56;
8     arr[2] = -7;
9     arr[3] = 0;
10    arr[4] = 5;
11    for (i=0; i<5; i++) {
12        printf("%d ", arr[i]);
13    }
14    return 0;
15 }

```

출력 예 22 56 -7 0 5

## 6.2 1차원 배열

소스 6-2 : 크기가 2인 int 형 배열 값 저장 방식의 이해

```

1 #include<stdio.h>
2 int main(void)
3 {
4     int arr[2];
5     arr[0] = 0;
6     arr[1] = 1;
7     printf("%d\n", arr);
8     printf("%d\n", &arr[0]);
9     printf("%d\n", &arr[1]);
10    return 0;
11 }

```

출력 예 350267704  
350267704  
350267708

## 6.2 1차원 배열

소스 6-3 : 배열 초기화 및 For 문을 이용한 출력

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int arr1[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
5     int arr2[10] = { 0 };
6     int arr3[] = { 1, 2, 3, 4, 5};
7     int i;
8     for (i = 0; i < 10; i++) {
9         printf("%d ", arr1[i]);
10    }
11    printf("\n");
12    for (i = 0; i < 10; i++) {
13        printf("%d ", arr2[i]);
14    }
15    printf("\n");
16    for (i = 0; i < 5; i++) {
17        printf("%d ", arr3[i]);
18    }
19    printf("\n");
20    return 0;
21 }

```

출력 예

```

1 2 3 4 5 6 7 8 9 10
0 0 0 0 0 0 0 0 0 0
1 2 3 4 5

```

## 6.2 1차원 배열

소스 6-4 : scanf() 함수를 이용하여 값을 입력 받아 배열의 원소에 저장

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int array[5];
5     int i, sum = 0;
6     for (i = 0; i < 5; i++) {
7         scanf("%d", &array[i]);
8         sum = sum + array[i];
9     }
10    printf("Total : %d", sum);
11    return 0;
12 }

```

입력 예

```
1 2 3 4 5
```

출력 예

```
Total : 15
```

## 6.2 1차원 배열

소스 6-5: while 문을 사용하여 배열의 원소에 접근

```

1 #include <stdio.h>
2 int main(void)
3 {
4     double arr[10];
5     double sum = 0;
6     int n = 0;
7     while(n != 10) {
8         scanf("%lf", &arr[n]);
9         if (arr[n] < 0) break;
10        sum = sum + arr[n];
11        n++;
12    }
13    printf("Average : %.1f\n", sum / n);
14    return 0;
15 }

```

입력 예 1 90.5 90 92 99.5 87.2 78.6 88 100 -5

출력 예 1 Average : 90.7

입력 예 2 100 99.5 88.1 77.7 66.6 99.9 95 94 93 92

출력 예2 Average : 90.6

## 6.2 1차원 배열

### ■ 배열을 이용한 선택 정렬 구현

#### ✓ 선택 정렬

##### ➤ 제자리 정렬 알고리즘

##### ➤ 최소값을 찾은 뒤 그 값을 맨 앞에 위치한 값과 교체

#### ✓ 입력 예시

입력	5	9	2	8	7	89	23	4	1	19
오름차순 정렬	1	2	4	5	7	8	9	19	23	89
내림차순 정렬	89	23	19	9	8	7	5	4	2	1



## 6.2 1차원 배열

1. 주어진 배열 내에서 가장 작은 값을 갖는 원소를 찾는다

✓ 가장 작은 수와 0번 인덱스의 값을 비교 후 비교한 숫자가 작다면 서로 교환

5	9	2	8	7	89	23	4	1	19
1	9	2	8	7	89	23	4	5	19

2. 0번 인덱스의 원소를 제외한 나머지 위치를 같은 방법으로 교체한다.

1	9	2	8	7	89	23	4	5	19
---	---	---	---	---	----	----	---	---	----

3. 배열에 원소 하나가 남을 때까지 1)에서 2)의 과정을 반복 한다.

1	2	4	5	7	8	9	19	23	89
---	---	---	---	---	---	---	----	----	----

## 6.2 1차원 배열

소스 6-6: 선택 정렬을 사용하여 배열의 각 원소의 값을 오름차순 출력

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int arr[10];
5     int i, j, min, tmp;
6     for (i = 0; i < 10; i++)
7         scanf("%d", &arr[i]);
8     for (i = 0; i < 9; i++) {
9         min = i;
10        for (j = i + 1; j < 10; j++) {
11            if (arr[j] < arr[min])
12                min = j;
13        }
14        tmp = arr[i];
15        arr[i] = arr[min];
16        arr[min] = tmp;
17    }
18    for (i = 0; i < 10; i++) {
19        printf("%d ", arr[i]);
20    }
21 }
```

입력 예 5 9 2 8 7 89 23 4 1 19

출력 예 1 2 4 5 7 8 9 19 23 89

## 6.2 1차원 배열

소스 6-7: 배열 안에서 원하는 값을 찾아보는 탐색

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int score[10] = {98, 80, 75, 77, 88, 99, 100, 65, 92, 84};
5     int i, index=-1, findScore;
6     printf("Input score for searching : ");
7     scanf("%d", &findScore);
8     for (i=0; i<10; i++) {
9         if(score[i]==findScore) {
10             index = i;
11             break;
12         }
13     }
14     if(index == -1) printf("There is no %d\n", findScore);
15     else printf("%d is in score[%d].\n", findScore, index);
16 }

```

입력 예 1	50
출력 예 1	Input score for searching : 50 There is no 50.
입력 예 2	99
출력 예 2	Input score for searching : 99 99 is in score[5].

## 6.2 1차원 배열

소스 6-8: 이진 탐색 예제

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int score[10] = {98, 80, 75, 77, 88, 99, 100, 65, 92, 84};
5     int i, j, max, tmp, top=0, btm=9, mid, findScore;
6     for (i=0; i<9; i++) {
7         max = i;
8         for (j=i+1; j<10; j++) {
9             if (score[j] > score[max])
10                 max = j;
11         }
12         tmp = score[i];
13         score[i] = score[max];
14         score[max] = tmp;
15     }
16     for (i=0; i<10; i++)
17         printf("%d ", score[i]);
18     printf("\nInput score for searching : ");
19     scanf("%d", &findScore);

```

## 6.2 1차원 배열

소스 6-8 이진 탐색 예제

```

20 while(1){
21     if(top<=btm){
22         mid=(top+btm)/2;
23         if(findScore==score[mid]){
24             printf("%d is %d's place.\n", findScore, mid+1,
25                 mid+1>2?"th":(mid+1==1?"st":"nd"));
26             break;
27         }
28         if(findScore<score[mid]) top=mid+1;
29         else btm=mid-1;
30     }else{
31         printf("There is no %d.\n", findScore);
32         break;
33     }
34 }

```

입력 예 1	50
출력 예 1	100 99 98 92 88 84 80 77 75 65 Input score for searching : 99 99 is 2nd place.
입력 예 2	99
출력 예 2	100 99 98 92 88 84 80 77 75 65 Input score for searching : 50 There is no 50.

## 6.2 1차원 배열

소스 6-9: 숫자 입력 횟수를 세어 출력 예제

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int histo[10] = {0};
5     int n=0, i, j;
6     while(1) {
7         scanf("%d", &n);
8         if(n<0) break;
9         else if(n<=9) histo[n]++;
10    }
11    for(i =0; i <10; i++) {
12        if(histo[i]==0) continue;
13        printf("%d (%d):", i, histo[i]);
14        for(j =0; j < histo[i]; j++)
15            printf("#");
16        printf("\n");
17    }
18 }

```

입력 예	5 9 8 7 89 23 4 1 19 0 0 0 7 7 -5
출력 예	0 (3):### 1 (1):# 4 (1):# 5 (1):# 7 (3):### 8 (1):# 9 (1):#

### 6.3 2차원 배열

#### ■ 2차원 배열

##### ✓ 정의

- 배열의 원소로 다른 1차원 배열을 가지는 배열

##### ✓ 선언 방법

- 1차원 배열 선언방법과 유사
- 배열의 크기 값이 2개 사용

배열 선언	데이터타입 배열이름[배열크기(행)][배열크기(열)];
배열 선언 예	<pre>int score[5][30]; double grade[2][10]; char name[30][15];</pre>

### 6.3 2차원 배열

#### ■ 2차원 배열의 구조

- ✓ arr[2][3] 배열 선언 시, 원소 인덱스에 따른 크기, 접근 방식 예제

주소	0000	0000	0004	0008
원소	arr[0]	arr[0][0]	arr[0][1]	arr[0][2]
크기	12	4	4	4
주소	0012	0012	0016	0020
원소	arr[1]	arr[1][0]	arr[1][1]	arr[1][2]
크기	12	4	4	4

### 6.3 2차원 배열

소스 6-10 : 2차원 배열 2개를 선언과 동시에 초기화 예제

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int arr1[3][3] = {{1, 2, 3},{4, 5, 6},{7, 8, 9}};
5     int arr2[3][3] = { 0 };
6     int i, j;
7     for (i = 0; i < 3; i++) {
8         for (j = 0; j < 3; j++)
9             printf("%d ", arr1[i][j]);
10        printf("\n");
11    }
12    for (i = 0; i < 3; i++) {
13        for (j = 0; j < 3; j++)
14            printf("%d ", arr2[i][j]);
15        printf("\n");
16    }
17 }

```

출력 예

```

1 2 3
4 5 6
7 8 9
0 0 0
0 0 0
0 0 0

```

### 6.3 2차원 배열

#### ■ 2차원 배열 선언 및 초기화

- ✓ 배열 선언 후 중괄호( { } ) 를 사용하여 원소 값 입력 시, 선언과 동시에 초기화 진행

arr1[0][0]	arr1[0][1]	arr1[0][2]
1	2	3
arr1[1][0]	arr1[1][1]	arr1[1][2]
4	5	6
arr1[2][0]	arr1[2][1]	arr1[2][2]
7	8	9

< 소스 6-10, 4번 라인 실행 결과 >

### 6.3 2차원 배열

소스 6-11: 2차원 배열 2개를 사용하여 행렬 곱셈을 수행하는 예제

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int mat1[2][2] = {{1, 2},{3, 4}};
5     int mat2[2][2] = {{5, 6},{7, 8}};
6     int resultMat[2][2] = {0};
7     int i, j, k;
8     for (i = 0; i < 2; i++)
9         for (j = 0; j < 2; j++)
10            for (k = 0; k < 2; k++)
11                resultMat[i][j] += mat1[i][k] * mat2[k][j];
12     for (i = 0; i < 2; i++) {
13         for (j = 0; j < 2; j++) {
14             printf("%d ", resultMat[i][j]);
15         }
16         printf("\n");
17     }
18 }

```

출력 예	19 22
	43 50

### 6.3 2차원 배열

#### ■ 2차원 배열 행렬 곱 예시

✓ 행렬 곱이란?

➢ 각 행렬의 값을 아래의 공식을 사용하여 곱셈을 수행

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj} = \sum_{k=1}^n a_{ik}b_{kj},$$

✓ 소스 6-11 동작 결과

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \times \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 1 \times 5 + 2 \times 7, & 1 \times 6 + 2 \times 8 \\ 3 \times 5 + 4 \times 7, & 3 \times 6 + 4 \times 8 \end{pmatrix} = \begin{pmatrix} 19, & 22 \\ 43, & 50 \end{pmatrix}$$

### 6.3 2차원 배열

소스 6-12: 2차원 배열을 이용한 성적 처리 프로그램 예제

```

1 #include <stdio.h>
2 int main(void)
3 {
4     int score[4][4] = {0};
5     int i, j;
6     for (i = 0; i < 4; i++) {
7         printf("Input 3 scores for student %d >> ", i+1);
8         for (j = 0; j < 3; j++) {
9             scanf("%d", &score[i][j]);
10            score[i][3] += score[i][j];
11        }
12    }
13    for (i = 0; i < 4; i++) {
14        printf("Total score of student %d is %d.\n", i+1,
15            score[i][3]);
16    }

```

입력 예	85 88 94
	95 100 92
	97 95 96
	77 82 86
출력 예	Input 3 scores for student 1 >> 85 88 94
	Input 3 scores for student 2 >> 95 100 92
	Input 3 scores for student 3 >> 97 95 96
	Input 3 scores for student 4 >> 77 82 86
	Total score of student 1 is 267.
	Total score of student 2 is 287
	Total score of student 3 is 288
	Total score of student 4 is 245

### 6.4 다차원 배열

#### ■ 3차원 배열

- ✓ 직육면체와 같은 입체 구조와 유사
- ✓ 2차원 배열은 행과 열로 구성되어 있다면, 3차원 배열은 면과 행, 열로 구성
- ✓ 선언 방법

배열 선언	데이터타입 배열이름[배열크기(면)][배열크기(행)][배열크기(열)];
배열 선언 예	<pre> int score[30][30][30]; double grade[10][10][10]; char name[15][15][15]; </pre>



## 6.4 다차원 배열

### 3차원 배열 구성

	arr3D[2][0][0]	arr3D[2][0][1]	arr3D[2][0][2]	
	arr3D[1][0][0]	arr3D[1][0][1]	arr3D[1][0][2]	
arr3D[0][0][0]	arr3D[0][0][1]	arr3D[0][0][2]		[2]
1	2	3		[2]
arr3D[0][1][0]	arr3D[0][1][1]	arr3D[0][1][2]		[2]
4	5	6		[2]
arr3D[0][2][0]	arr3D[0][2][1]	arr3D[0][2][2]		
7	8	9		

## 6.4 다차원 배열

### 3차원 배열의 구성

주소	0000	0000	0000	0004	0008
원소	arr3D[0]	arr3D[0][0]	arr3D[0][0][0]	arr3D[0][0][1]	arr3D[0][0][2]
크기	36	12	4	4	4
주소		0012	0012	0016	0020
원소		arr3D[0][1]	arr3D[0][1][0]	arr3D[0][1][1]	arr3D[0][1][2]
크기		12	4	4	4
주소		0024	0024	0028	0032
원소		arr3D[0][2]	arr3D[0][2][0]	arr3D[0][2][1]	arr3D[0][2][2]
크기		12	4	4	4
주소	0036	0036	0036	0040	0044
원소	arr3D[1]	arr3D[1][1]	arr3D[1][1][0]	arr3D[1][1][1]	arr3D[1][1][2]
크기	36	12	4	4	4
주소		0048	0048	0052	0056
원소		arr3D[1][1]	arr3D[1][1][0]	arr3D[1][1][1]	arr3D[1][1][2]
크기		12	4	4	4
주소		0060	0060	0064	0068
원소		arr3D[1][2]	arr3D[1][2][0]	arr3D[1][2][1]	arr3D[1][2][2]
크기		12	4	4	4
주소	0072	0072	0072	0076	0080
원소	arr3D[2]	arr3D[2][1]	arr3D[2][1][0]	arr3D[2][1][1]	arr3D[2][1][2]
크기	36	12	4	4	4
주소		0084	0084	0088	0092
원소		arr3D[2][1]	arr3D[2][1][0]	arr3D[2][1][1]	arr3D[2][1][2]
크기		12	4	4	4
주소		0096	0096	0100	0104
원소		arr3D[2][2]	arr3D[2][2][0]	arr3D[2][2][1]	arr3D[2][2][2]
크기		12	4	4	4



## 6.4 다차원 배열

소스 6-12: 3차원 배열을 활용한 학급 별 성적 관리 프로그램 예시

```

1 #include<stdio.h>
2 int main(void)
3 {
4     int score[3][3][3], total[3][3]={0};
5     int i, j, k;
6     for (i = 0; i < 3; i++)
7         for (j = 0; j < 3; j++) {
8             printf("%d학년 %d반 세 학생의 성적을 입력하시오 : ", i+1,
9 j+1);
10             for (k = 0; k < 3; k++) {
11                 scanf("%d", &score[i][j][k]);
12                 total[i][j] += score[i][j][k];
13             }
14             for (i = 0; i < 3; i++)
15                 for (j = 0; j < 3; j++)
16                     printf("%d학년 %d반의 총점은 %d점 입니다.\n", i+1,
17 j+1, total[i][j]);
18 }

```

School of Computer Software at Dae

입력 예

70 80 80  
90 60 60  
40 40 40  
70 20 90  
40 40 70  
30 30 60  
10 90 90  
20 70 80  
70 70 70

출력 예

1학년 1반 세 학생의 성적을 입력하시오 : 70 80 80  
1학년 2반 세 학생의 성적을 입력하시오 : 90 60 60  
1학년 3반 세 학생의 성적을 입력하시오 : 40 40 40  
2학년 1반 세 학생의 성적을 입력하시오 : 70 20 90  
2학년 2반 세 학생의 성적을 입력하시오 : 40 40 70  
2학년 3반 세 학생의 성적을 입력하시오 : 30 30 60  
3학년 1반 세 학생의 성적을 입력하시오 : 10 90 90  
3학년 2반 세 학생의 성적을 입력하시오 : 20 70 80  
3학년 3반 세 학생의 성적을 입력하시오 : 70 70 70  
1학년 1반의 총점은 230점 입니다.  
1학년 2반의 총점은 210점 입니다.  
1학년 3반의 총점은 120점 입니다.  
2학년 1반의 총점은 180점 입니다.  
2학년 2반의 총점은 150점 입니다.  
2학년 3반의 총점은 120점 입니다.  
3학년 1반의 총점은 190점 입니다.  
3학년 2반의 총점은 170점 입니다.  
3학년 3반의 총점은 210점 입니다.

## 6.5 Q&A

Q 1. Q 이 코드는 왜 동작하지 않는가?

```
a[i] = i++;
```

A. 부분식인 'i++'은 에러를 발생 시킬 수 있다, 배열의 인덱스를 가리키는 i의 값 또한 변경되기 때문에 이 값이 변경되기 이전의 값일지, 변경된 다음의 값일지 알 수가 없다.

Q 2. 다음 코드는 맞는 표현인가?

```
char a[3] = "abc";
```

A. ANSI C는 올바른 표현이라고 말하지만 실제로 사용되지는 않는다. 해당 코드는 3개의 요소를 갖는 배열을 선언하고 각각을 'a', 'b', 'c'로 초기화한다. 뒷부분에서 문자열에 관해 배우겠지만 여기에서는 C에서 문자열을 나타낼 때 사용되는 '₩0' 문자가 없으므로 문자열이라고 말하기에는 곤란하다. 따라서 strcpy() 함수 또는 printf() 함수의 인자로 전달될 수 없다.

## 6.5 Q&A

Q 3. 배열의 크기를 지정할 때, 다음과 같이 상수 값을 쓰면 안되는 이유가 무엇인가?

```
const int n =5;
```

```
int a[n];
```

A. const 지정자는 “읽기 전용”을 의미한다. 즉 지정한 오브젝트는 실행할 때 변경할 수 없는 run-time 오브젝트이다. 따라서 이러한 'const' 오브젝트는 상수 수식(constant expression)이 아니기 때문에, 배열의 크기 지정, case label 등에 쓰일 수 없다. 정말로 compile-time 상수가 필요하다면, #define으로 (또는 enum으로) 상수를 정의하여라

## 6.6 실습 및 과제

- 실습 및 과제 진행
  - DCU Code : <http://code.cu.ac.kr>
- 6 주차 실습
  - 코딩 6-1, 코딩 6-2, 코딩 6-3, 코딩 6-4, 코딩 6-5, 코딩 6-6 (p243 ~ 254)
- 6 주차 과제
  - 6장 프로그래밍 연습 1-5번 (p261-262)



## 6.7 참고 문헌

---

- 배열 : <https://ko.wikipedia.org/wiki/배열>
- C Programing FAQs : <http://cinsk.github.io/ko/cfaqs/index.html>

END

