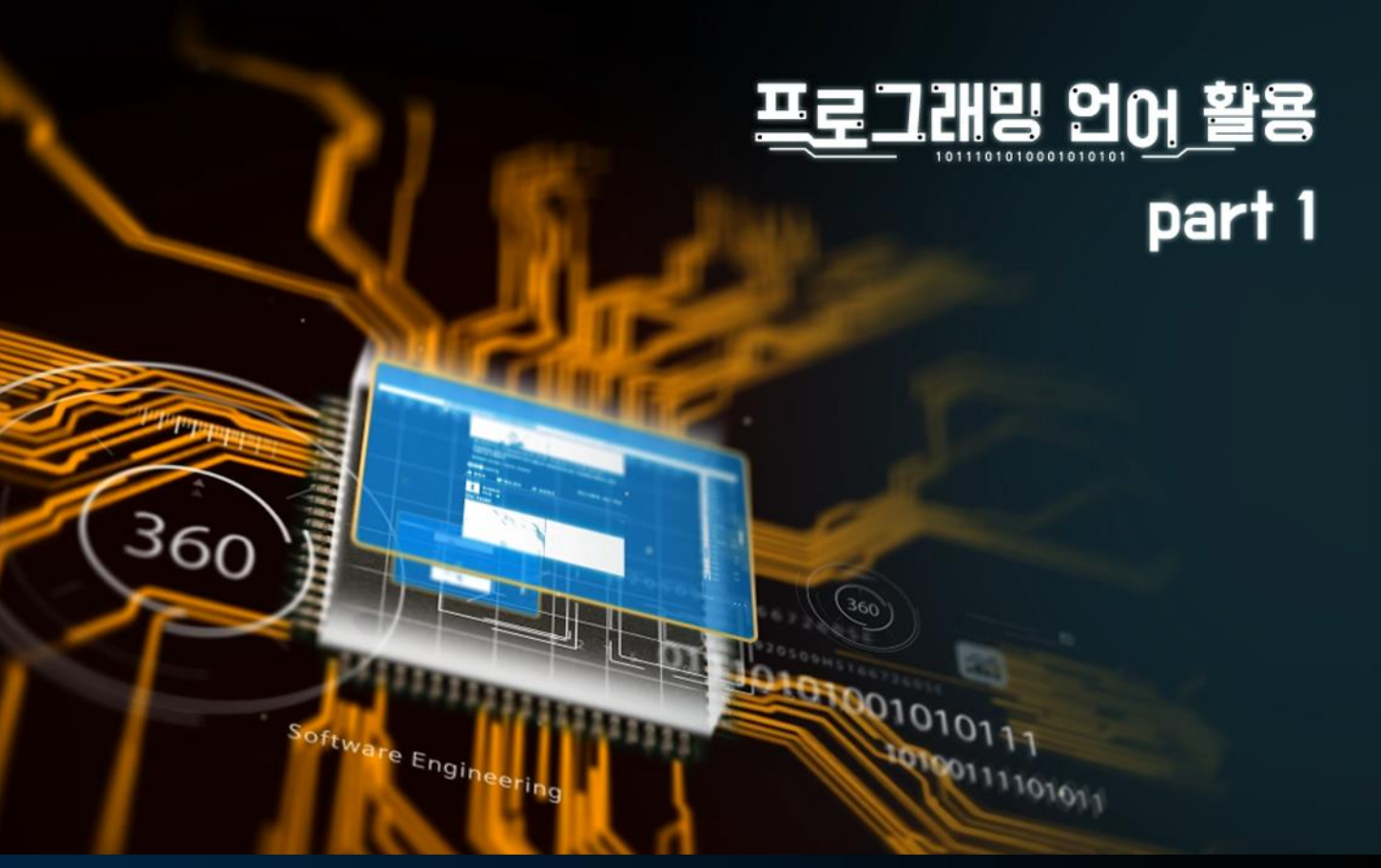


포인터와 문자열

한국기술교육대학교
온라인평생교육원
OLEI

학습내용

- 포인터 배열
- 배열 포인터

학습목표

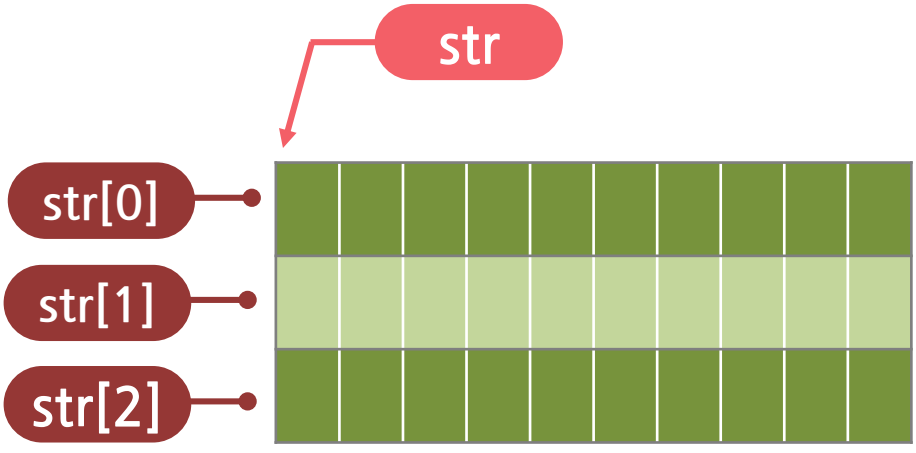
- 2차원 배열과 포인터 배열에 대해 설명할 수 있다.
- 포인터를 이용하여 문자열을 처리할 수 있다.
- 배열 포인터의 개념과 용도에 대해 설명할 수 있다.

포인터와 배열

1 2차원 배열과 포인터

str[3][10]

str = str[0] = &str[0][0]



str : 이중 포인터

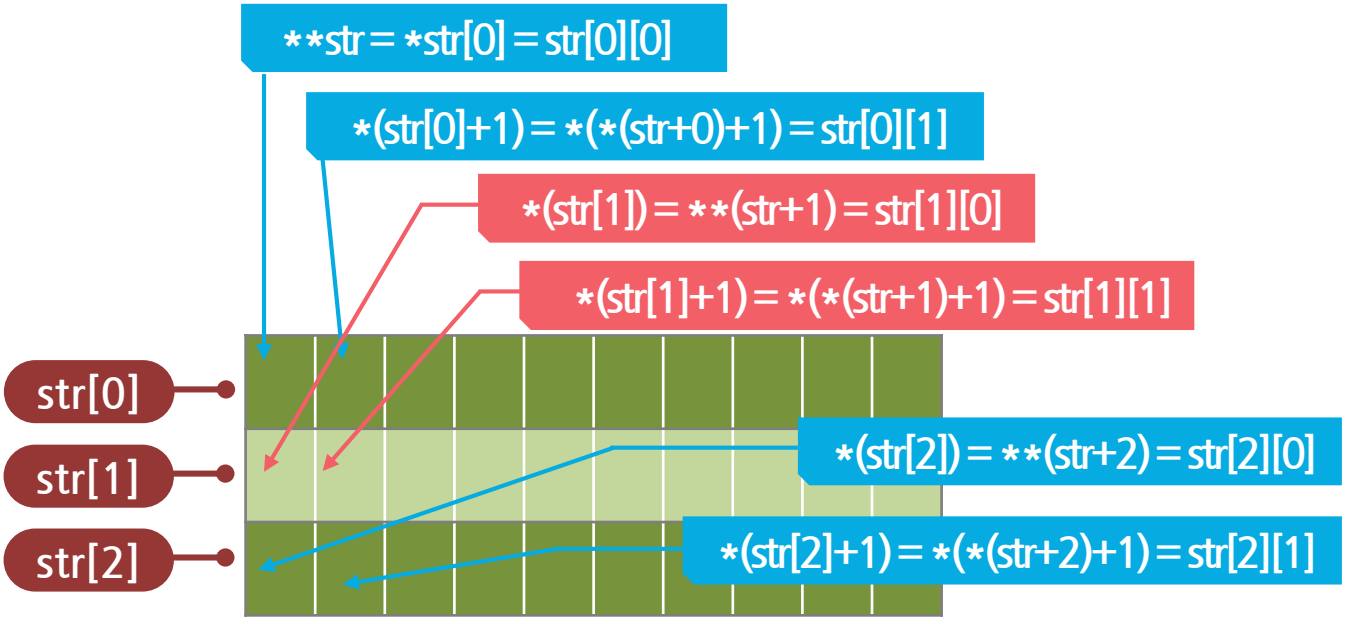
**str = *str[0] = str[0][0]

a[2][3]

a[0][0]	*(*a + 0)	*(a[0] + 0)	*(*(a+0) + 0)
a[0][1]	*(*a + 2)	*(a[0] + 1)	*(*(a+0) + 1)
a[0][2]	*(*a + 3)	*(a[0] + 2)	*(*(a+0) + 2)
a[1][0]	*(*a + 4)	*(a[1] + 0)	*(*(a+1) + 0)
a[1][1]	*(*a + 5)	*(a[1] + 1)	*(*(a+1) + 1)
a[1][2]	*(*a + 6)	*(a[1] + 2)	*(*(a+1) + 2)

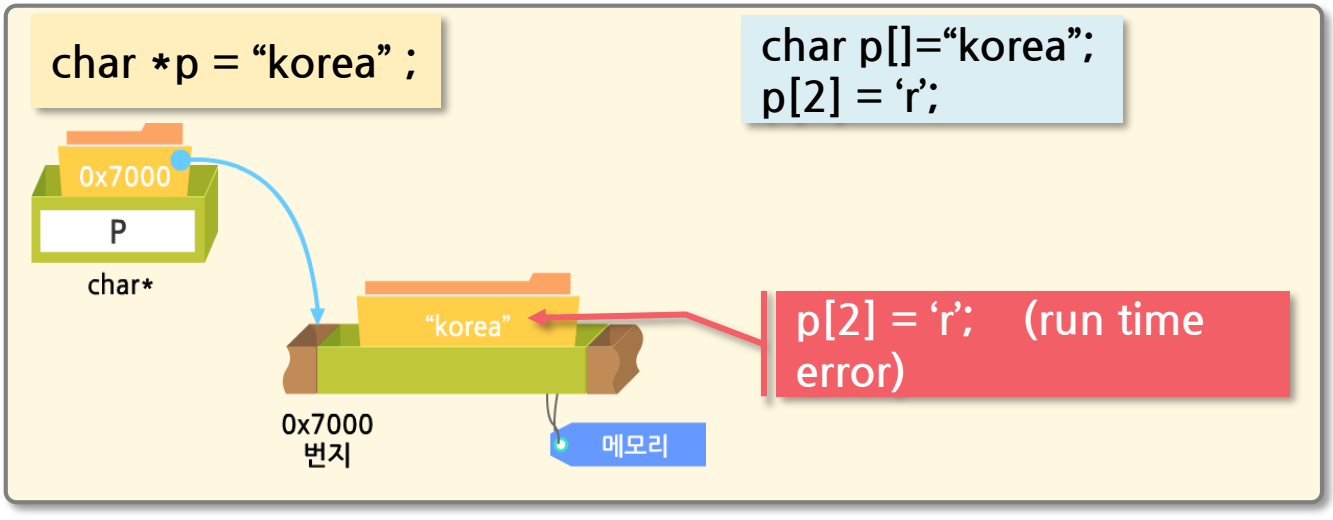
포인터와 배열

1 2차원 배열과 포인터



2 문자형 포인터

문자열 상수를 문자형 포인터 변수에 저장



포인터와 배열

2 문자형 포인터

출력

```
puts(p);
```

```
printf("%s ₩n", p);
```

p[i]

```
int i = 0 ;
while ( p[i] ! = '₩0' )
    printf( "%c", p[i] ) ;
printf("₩n" );
```

```
char a[] = "Hello World!";
```

H	e	l	l	o		w	o	r	l	d	!	\0	\0	\0	\0

```
a[5] = '₩0';    ₩0
```

```
printf("%s   %s", a, a+6);
```

포인터와 배열

2 문자형 포인터

char *str = “good morning”;에서 str의 문자열을 거꾸로 출력하는 프로그램

```
#include <stdio.h>

int main()
{
 char *str="good morning";
 int i,count=0;
 while(str[i]){
  count++;
}
for(i=count-1;i>=0;i--)
   printf("%c",*(str+i));

  return 0;
}
```

배열 포인터

1 배열 포인터 개요

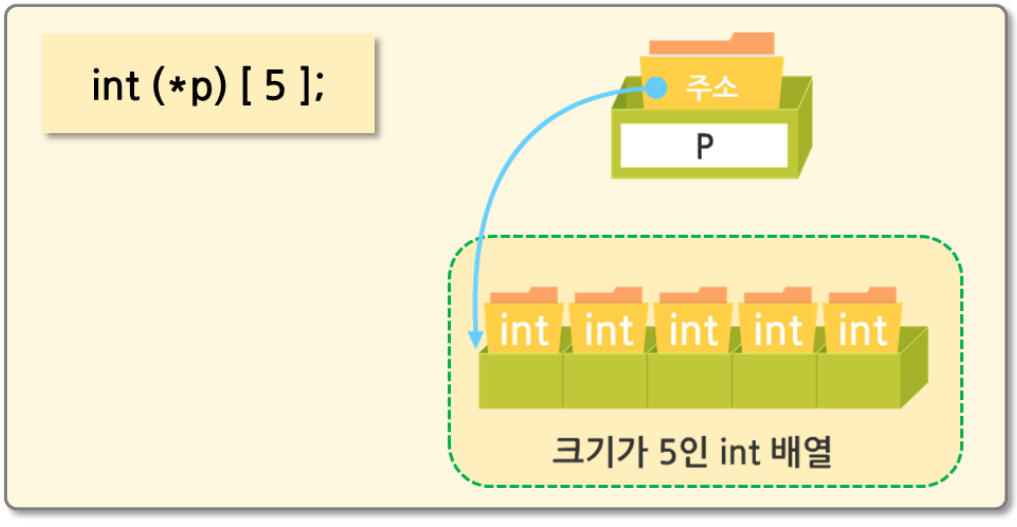
배열 포인터는 이차원 배열의 전체를 가리키는 용도로 사용

형식 데이터형 (*포인터명)[배열 크기];

예제

```
int       (*p1) [ 5 ];   → int[5] 배열을 가리키는 포인터
char      (*p2) [10 ];   → char[10] 배열을 가리키는 포인터
double    (*p3) [ 4 ];   → double[4] 배열을 가리키는 포인터
STUDENT   (*p4) [ 3 ];   → STUDENT[3] 배열을 가리키는 포인터
```

int (*p) [5]; ——— 크기가 5인 int 배열을 가리키는 포인터

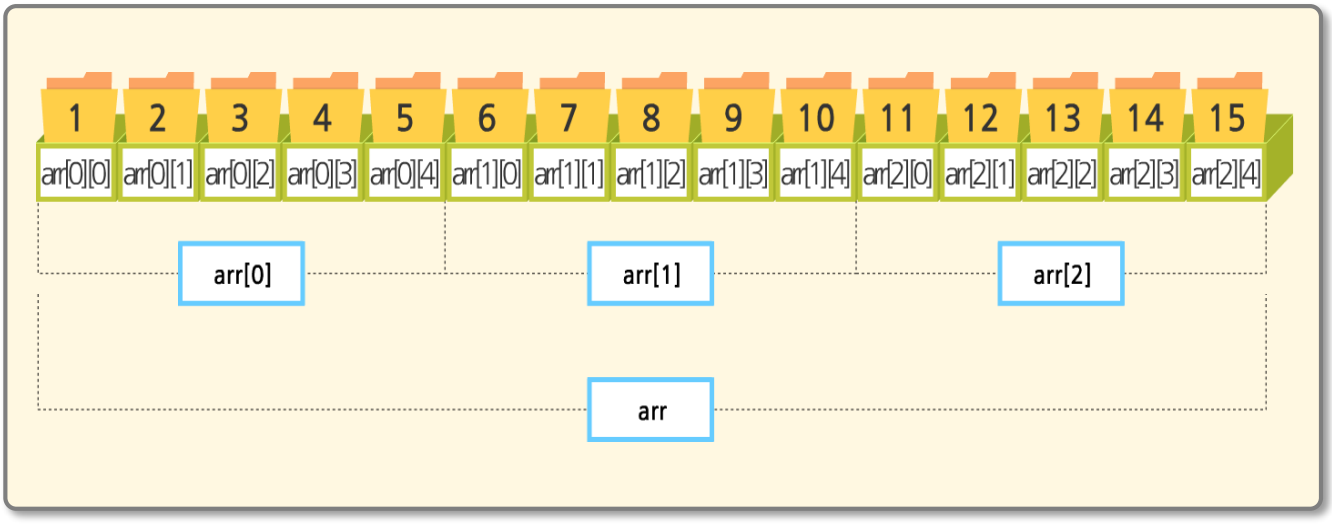


배열 포인터

1 배열 포인터 개요

```
int arr[3][5] = {
    { 1, 2, 3, 4, 5 },
    { 6, 7, 8, 9, 10},
    {11, 12, 13, 14, 15}
};
int (*p)[5] = &arr[0];
```

p를 int 5개 짜리 배열인 arr[0]의 주소로 초기화함



1 배열 포인터를 &arr[0]으로 초기화하는 대신, 간단하게 arr로 초기화 가능

```
int (*p) [ 5 ] = arr;
```

arr는 &arr[0]과 같은 의미임

배열 포인터

1 배열 포인터 개요

2 배열 포인터 p로 2차원 배열처럼 참조 가능

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

p[i][j]는 이차원 배열의 원소 arr[i][j]를 의미함

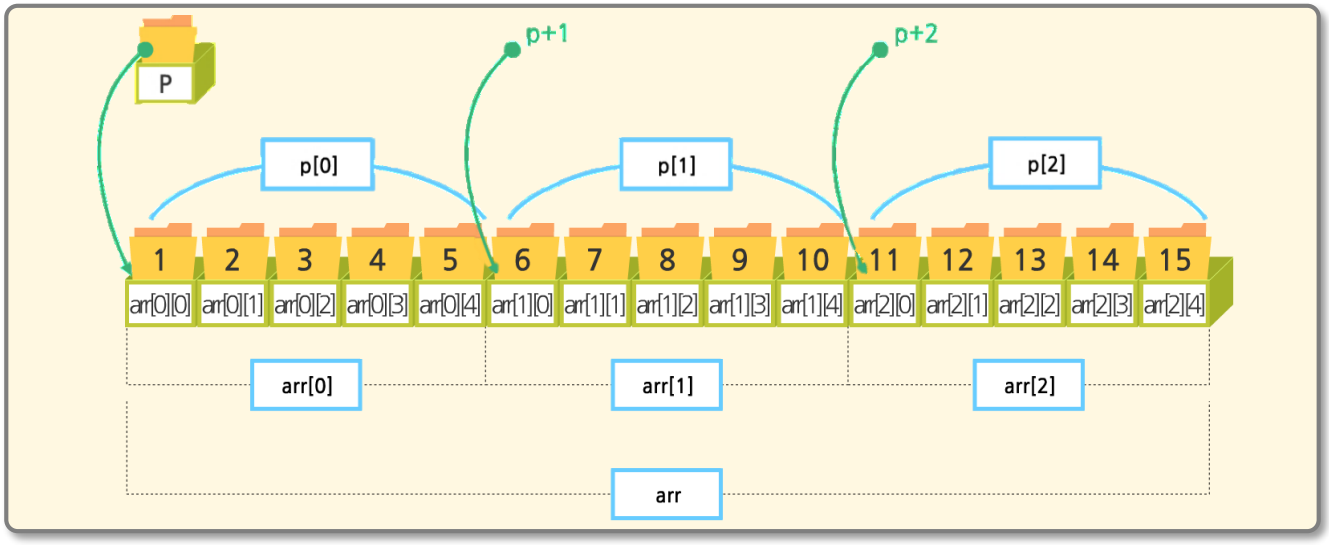
3 배열 포인터를 이차원 배열에 접근하기 위한 용도로 사용

```
int arr[3][5]
int (*p)[5] = arr;
```

*(*(p + i) + j) = = p [i] [j]

arr[i]

arr[i][j] arr[i][j]



배열 포인터

1 배열 포인터 개요

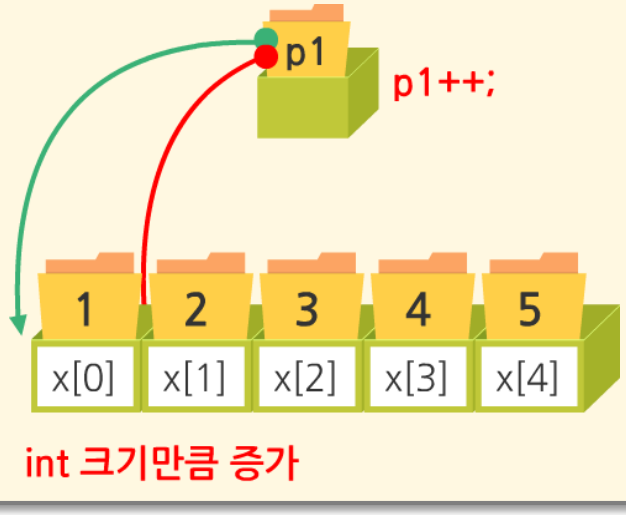
4 2차원 배열에 대한 배열 포인터는 열 크기에 따른 변수 선언이 필요

	1차원 배열 포인터	2차원 배열 포인터
선언	데이터 타입 * 포인터 변수명;	데이터 타입 (* 포인터 변수명)[크기];
초기화	변수명 = 배열명; 데이터 타입 *변수명 = 배열명;	변수명 = 배열명; 데이터 타입 *변수명[크기] = 배열명;

예	int a[3]={1,2,3};
	int *p=a;
	p = a;
		int a[2][3]={{1,2,3},{4,5,6}};
		int (*p)[3]=a;
		p = a;

5 배열 포인터의 증감 연산은 열 크기만큼 이동

```
int x[5]
int *p1 = x;
p1++;
```



배열 포인터

1 배열 포인터 개요

5 배열 포인터의 증감 연산은 열 크기만큼 이동

```
int y[2][3];
int (*p2)[3] = y;
p2++;
```

int[3] 크기만큼 증가

배열 포인터

2 문자 포인터 배열

1 여러개의 문자열을 하나로 묶어서 처리하는 방법으로 2차원 문자배열을 이용

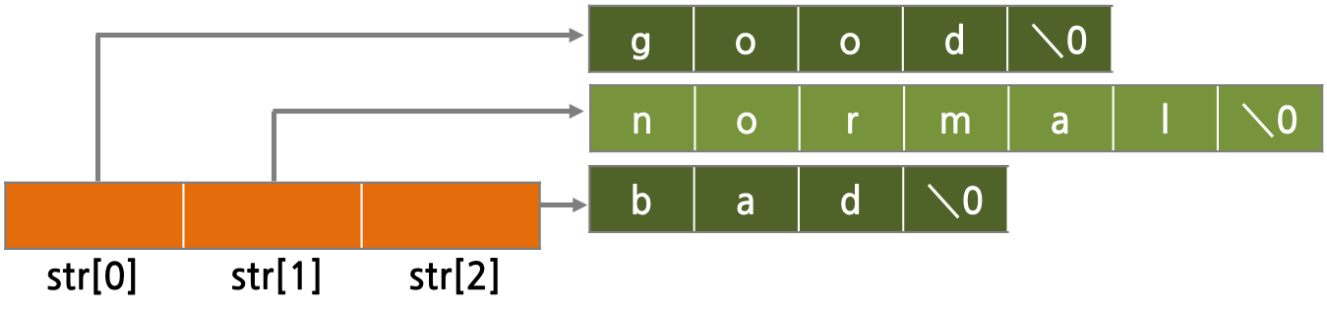
```
char str[][6]={ “good”, “nomal”, “bad” };
```

g	o	o	d	\0	\0	\0
n	o	r	m	a	l	\0
b	a	d	\0	\0	\0	\0

```
printf(“%s  %s   %s  ”, str[0], str[1], str[2] );
```

2 여러개의 문자열을 하나로 묶어서 처리하는 방법으로 문자포인터배열을 이용

```
char *str[]={ “good”, “nomal”, “bad” };
```




```
printf(“%s  %s   %s  ”, str[0], str[1], str[2] );
```

학습정리

1. 포인터 배열

- 포인터 배열은 주소를 보관하는 배열임
- 포인터 배열을 이용하여 각 변수를 참조하는 것이 가능함
- 여러 개의 문자열을 하나로 묶어 처리하는 방법으로 문자 포인터 배열을 이용할 수 있음

2. 배열 포인터

- 배열 포인터란 배열 전체를 가리키는 포인터로 사용됨
- 배열 포인터 선언 형식은 “데이터 타입 (*포인터 변수명)[크기];”와 같은 형식으로 선언함