



Chapter 07. 포인터 고급

목차

1. 함수의 매개변수로 사용하는 배열
2. 문자열과 포인터
3. 함수를 가리키는 포인터

학습목표

- 배열을 매개변수로 하는 함수를 작성할 수 있다.
- 문자열과 포인터 사이의 관계를 말할 수 있다.
- 명령행에서 문자열을 읽어오는 방법을 학습한다.
- 함수를 가리키는 포인터를 사용할 수 있다.

01 함수의 매개변수로 사용하는 배열

■ 함수를 매개변수로 사용하는 1차원 배열

- 함수를 호출할 때 배열에 저장된 원소를 한꺼번에 넘겨주지 못하므로 1차원 배열의 시작 주소만 전달한다.
그러므로 형식 매개변수는 주소를 저장하기 위해 1차원 포인터 변수가 선언되어야 한다.

■ 함수의 매개변수로 사용하는 2차원 배열

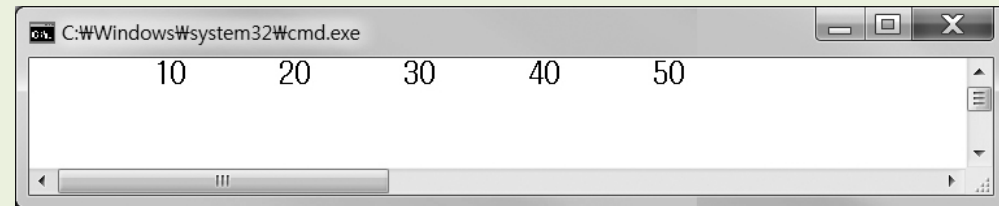
- 2차원 배열은 매개변수로 2차원 배열 전체를 넘겨주지 못하므로 2차원 배열의 시작 주소만 전달한다.
따라서, 이 주소를 2차원 포인터 변수로 받는다.

```
int (*p)[배열의 원소 개수];
```

포인터 변수 기본 형식

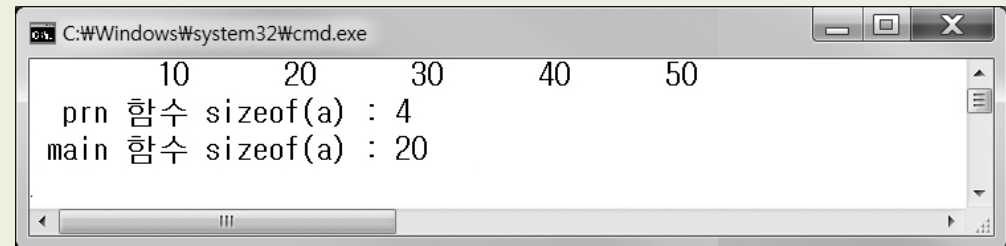
예제 7-1. 1차원 배열의 원소를 출력하는 함수 사용하기(07_01.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void prn(int *pa, int size);
04 void main()
05 {
06     int a[5] = {10,20,30,40,50};
07     prn(a, 5);
08 }
09 void prn(int *pa, int size)
10 {
11     for(int i=0; i<size; i++) {
12         cout<<"\t"<<*(pa+i);    // pa[i]와 같이 표현할 수 있다.
13     }
14     cout<<"\n";
15 }
```



예제 7-2. 포인터 매개변수를 배열로 표현하기(07_02.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void prn(int a[], int size);
04 void main()
05 {
06     int a[5] = {10,20,30,40,50};
07     prn(a, 5);
08     cout << " main 함수 sizeof(a) : "<< sizeof(a) << endl;
09 }
10 void prn(int a[], int size)    // void prn(int *a, int size)와 동일함
11 {
12     for(int i = 0;i<size;i++)
13         cout<<"\t"<<a[i];
14     cout<< endl;
15     cout << " prn 함수 sizeof(a) : "<< sizeof(a) << endl;
16 }
```



```
C:\Windows\system32\cmd.exe
10    20    30    40    50
prn 함수 sizeof(a) : 4
main 함수 sizeof(a) : 20
```

예제 7-3. 2차원 배열을 전달받는 함수 작성하기(07_03.cpp)

```
#include <iostream>
using namespace std;
#define ROW 3
#define COL 4
```

```
void prn(int (*p)[COL]);
```

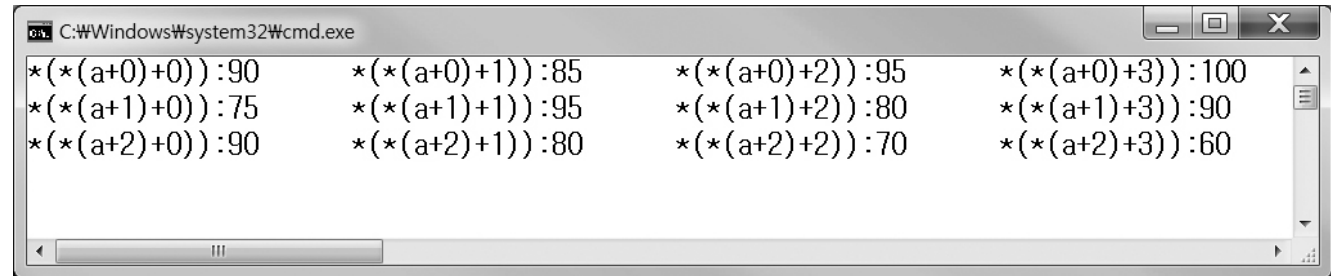
```
void main()
```

```
{
    int a[ROW][COL] = { { 90, 85, 95, 100 },
                        { 75, 95, 80, 90 },
                        { 90, 80, 70, 60 }
                      };

    prn(a);
}
```

```
void prn(int (*p)[COL])    // 2차원 배열의 주소값을 전달할 포인터
```

```
{
    int r, c;
    for (r = 0; r < ROW; r++) {
        for (c = 0; c < COL; c++) {
            cout << "*(a+" << r << ")+" << c << "):" << (*(p + r) + c) << " ";
        }
        cout << "\n";
    }
}
```



```
C:\Windows\system32\cmd.exe
*(*(a+0)+0):90      *(*(a+0)+1):85      *(*(a+0)+2):95      *(*(a+0)+3):100
*(*(a+1)+0):75      *(*(a+1)+1):95      *(*(a+1)+2):80      *(*(a+1)+3):90
*(*(a+2)+0):90      *(*(a+2)+1):80      *(*(a+2)+2):70      *(*(a+2)+3):60
```

02 문자열과 포인터

■ 문자열 저장 방식

- ① 배열을 사용한다.

```
char str[10]="fox"; // 문자 배열
```

- 문자 배열에 저장하게 되면 메모리가 할당되어 문자열을 직접 저장한다.

str[0]	str[1]	str[2]	str[3]	str[4]	str[5]	str[6]	str[7]	str[8]	str[9]
'f'	'o'	'x'	NULL						

- ② 포인터 변수가 저장하는 형태다.

```
char *ptr="fox"; //포인터 변수
```

- 포인터 변수는 메모리에 저장된 문자열 상수의 시작주소이다.

ptr(문자열이 저장된 메모리의 시작 주소를 저장함)



(메모리상의 어딘가에 문자열이 저장되어 있음)

- 시작 주소만 저장하고 있는 포인터 변수 ptr로 다음과 같이 문자열을 출력할 수 있다.

```
cout << ptr;
```


02 문자열과 포인터

■ 포인터 변수에 문자열 상수 대입하기

- str이란 문자 배열을 선언하면서 "Apple"을 초기값으로 준 것이다.

```
char str[256]="Apple"; // 문자 배열 선언 시 초기화
```

- 다음은 이미 선언된 문자 배열에 문자열 상수를 대입하는 예다. 문자 배열에 문자열 상수를 대입연산자로 저장하는 것은 불가능하므로 다음과 같은 컴파일 에러가 발생한다.

```
str="Grapes"; //컴파일 에러 발생
```

02 문자열과 포인터

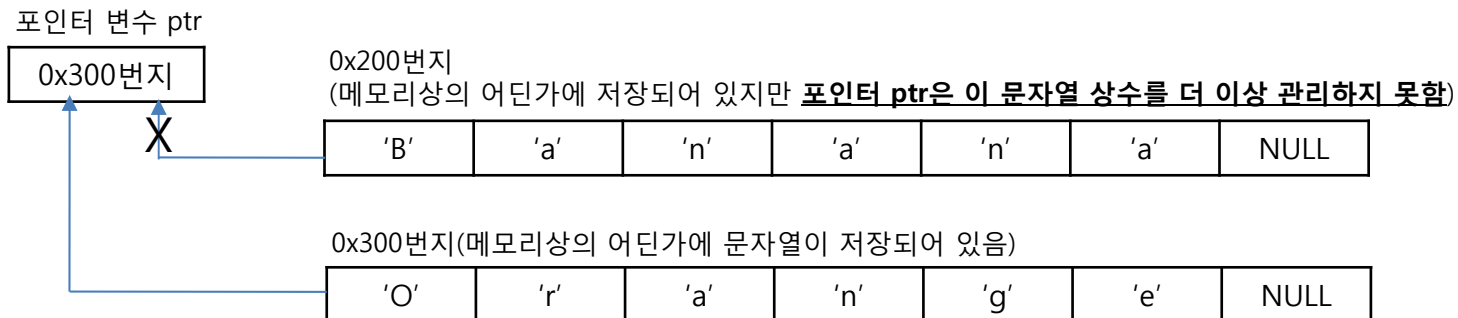
- 문자열 상수를 대입 연산자로 저장하려면 포인터 변수를 사용해야만 한다. 다음은 포인터 변수 ptr을 선언하면서 "Banana"를 초기값으로 준 것이다.

`char *ptr="Banana";` // 포인터 변수 선언 시 초기화



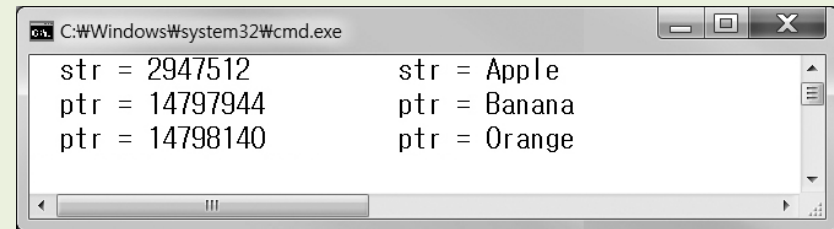
- 문자열 상수 "Orange"를 ptr에 대입하면 ptr은 더 이상 문자열 상수 "Banana"의 시작 주소값을 기억하지 못하고 문자열 상수 "Orange"를 가리키게 된다.

`ptr="Orange";`



예제 7-6. 포인터 변수에 문자열 대입하기(07_06.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     char str[256] = "Apple";
06     char *ptr = "Banana";
07
08     cout<<" str = "<<(int)str<<" Wt str = "<<str<<"\n";
09     cout<<" ptr = "<<(int)ptr<<" Wt ptr = "<<ptr<<"\n";
10
11     // str="Grapes"; // 문자 배열은 다른 문자열 상수를 대입하지 못한다.
12
13     // 포인터 변수에는 다른 문자열 상수를 대입할 수 있다.
14     ptr = "Orange";
15     // 포인터 변수 ptr에 다른 주소가 저장되어 있다.
16     cout<<" ptr = "<<(int)ptr<<" Wt ptr = "<<ptr<<"\n";
17 }
```



```
C:\Windows\system32\cmd.exe
str = 2947512      str = Apple
ptr = 14797944    ptr = Banana
ptr = 14798140    ptr = Orange
```

02 문자열과 포인터

■ 함수의 매개변수로 문자열 여러 개를 전달하기

- 문자열 여러 개를 저장하는 포인터 배열

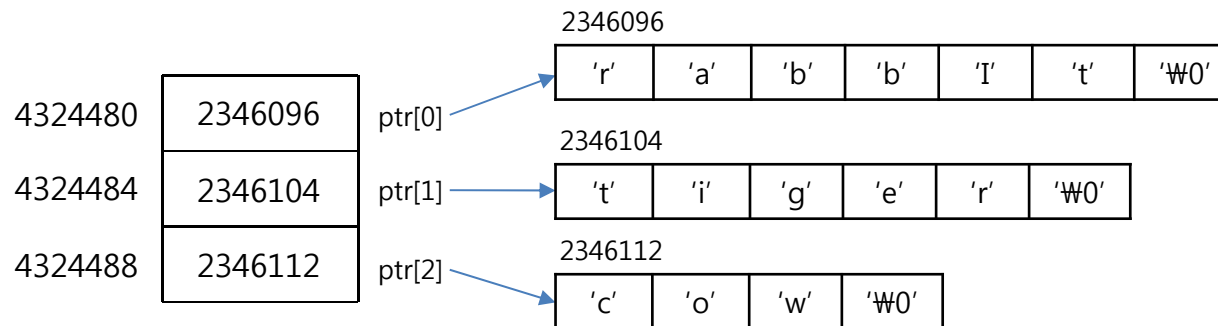
```
char *ptr[3];
```

- 문자열 상수를 각 배열의 요소에 대입하면 배열의 요소에는 문자열 상수의 시작 주소를 저장하게 된다.

```
ptr[0]="rabbit";
```

```
ptr[1]="tiger";
```

```
ptr[2]="cow";
```

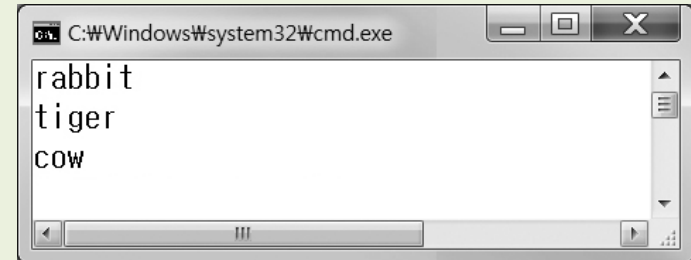


- 포인터 배열을 전달받는 함수
 - 포인터 배열의 시작 주소는 2차원 포인터이므로 포인터 배열을 매개변수로 하는 함수의 형식 매개변수 자료형은 2차원 포인터 형태여야 한다. 그리고 포인터 배열의 요소에 대한 개수도 전달해주어야 한다.

```
void print_string(char **pptr, int n)
```

예제 7-8. 포인터 배열을 매개변수로 사용하는 함수(07_08.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void print_string(char **pptr, int n)
04 {
05     for(int i=0; i<n ; i++)
06         cout<< pptr[i]<<" \n" ;    // *(pptr+i)
07 }
08 void main()
09 {
10     char *ptr[3]={"rabbit", "tiger", "cow"};
11     print_string(ptr, 3);
12 }
```



03 함수를 가리키는 포인터

- 메모리 영역은 변수를 위한 **데이터 블록**과 소스코드를 위한 **코드 블록**이 존재한다. 프로그램이 실행되면 각 함수의 코드는 코드 블록의 특정 번지에 저장된다. 함수가 호출되면 그 주소로 분기했다가 함수 호출문 다음으로 되돌아온다. 함수에 대한 포인터 변수는 메모리 영역에 저장된 함수의 시작 주소를 갖는다.

```
int (*pf)(int);    // 함수 포인터 선언
```

- pf는 함수의 주소값을 담는 포인터 변수로서, pf가 가리키는 함수는 정수 형태의 매개변수 하나를 갖고, int형을 결과값으로 되돌린다.
- 함수의 주소를 알아내려면 함수명만 기술한다. 절댓값을 구하는 abs() 함수가 다음과 같이 정의되어 있다고 해 보자.

```
int abs(int num)
{
    if(num<0)
        num=-num;
}
```

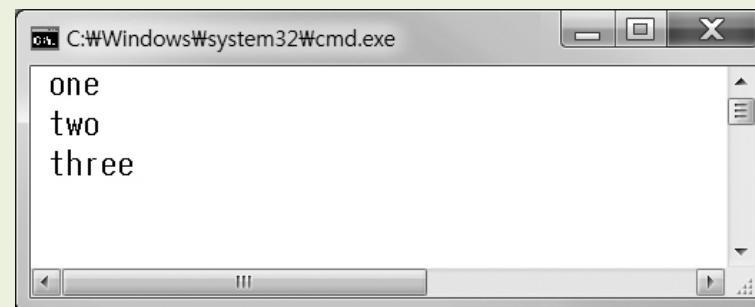
- 특정 함수의 주소를 함수 포인터에 저장한 후 포인터를 사용해서 함수 호출하는 방법은 다음과 같다.

```
pf = abs;    // 함수 포인터
int y = pf(-5);
```

예제 7-9. 함수를 포인터를 사용해서 함수 호출하기(07_09.cpp)

```
01 #include <iostream>
02 using namespace std;
03 /* 함수를 가리키는 포인터 변수 선언 */
04 void (*pf)(void);
05 void one()
06 {
07     cout<<" one \n";
08 }
09 void two()
10 {
11     cout<<" two \n";
12 }
13 void three()
14 {
15     cout<<" three \n";
16 }
17 void main()
18 {
19     pf = one;
20     pf();
21
22     pf = two;
23     pf();
```

```
24
25     pf = three;
26     pf();
27 }
```



Homework

- Chapter 7 Exercise: 2, 3, 4, 6