C언어 (CSE2035)
(Chap9. Pointer Applications) (2-1)

Sungwon Jung, Ph.D.

Dept. of Computer Science and Engineering
Sogang University
Seoul, Korea
Tel: +82-2-705-8930
Email : jungsung@sogang.ac.kr

Passing an array to a function

- 배열을 인자로 하는 함수의 일반적인 호출 형태는 다음과 같다.
  - 배열의 시작 주소값을 인자로 전달한다.

```
doIt (aryName);
```

- 따라서 배열을 인자로 받는 함수의 헤더는 다음과 같을 것이다.

```
int doIt (int ary[])
```

- 또한 1차원 배열을 인자로 하는 경우 포인터를 이용한 경우도 가능하다.

```
int doIt (int *arySalary)
```

Passing an array to a function

- 함수의 인자로 배열명을 전달하면, 배열의 시작 주소가 전달된다.

```
doIt (aryName);
```

배열명은 배열의 시작 주소값을 가지므로 주소가 전달된다.

- 호출되는 함수의 헤더에서는 형식 매개변수(formal parameter)로 포인터 변수를 사용하여야 한다.

```
int doIt (int *arySalary)
```

- 표기상의 편리성 때문에 포인터를 매개변수로 선언할 때 배열의 각괄호 표기법을 사용할 수 있다.
  - 이 때 배열로 선언된 형식 매개변수는 실질적으로는 포인터이다.
  - 배열 원소 자체는 복사되지 않는다.

```
int doIt (int ary[])
```

→ 사실상 포인터!

Passing an array to a function

- 예제 프로그램 – a[]와 *a가 같음을 보여주는 예제

```c
#include <stdio.h>

double sum1(double a[], int n);
double sum2(double *a, int n);

int main(void)
{
        double a[6]={1, 2, 3, 4, 5, 6};

        printf("sum1 = %f\n", sum1(a,6));
        printf("sum2 = %f\n", sum2(a,6));

        return 0;
}

double sum1(double a[], int n)
{

        int i;
        double sum=0.0;
        for(i=0; i<n; ++i)
                sum+=a[i];
        return sum;
}

```

```c
double sum2(double *a, int n)
{
        int i;
        double sum = 0.0;
        for(i=0; i<n; i++)
                sum+=a[i];
        return sum;
}

```

배열의 주소값과 배열의 길이를 인자로 전달

```
[root@mclab chap10]# ./chap10-4
sum1 = 21.000000
sum2 = 21.000000
[root@mclab chap10]#
```

두 함수의 수행 결과는 서로 같다.

형식 파리미터가 서로 다르다.

Passing an array to a function

- 배열을 인자로 하여 앞의 **sum()** 함수를 호출했을 때 계산되는 값은 다음 표와 같다. **(v**는 **main** 함수에서 선언된 배열이라 가정한다.**)**

호출	계산 및 리턴되는 값
sum(v, 100)	v[0] + v[1] + … + v[99]
sum(v, 88)	v[0] + v[1] + … + v[87]
sum(&v[7], k - 7)	v[7] + v[8] + … + v[k - 1]
sum(v + 7, 2 * k)	v[7] + v[8] + … + v[2 * k + 6]

Passing an array to a function

- 2차원 이상의 배열을 형식 매개변수로 갖는 함수는 (첫 번째를 제외한 나머지 차원의) 배열 크기를 알고 있어야 한다.
  - 예) 다음은 3차원 배열을 형식 매개변수로 갖는 함수의 헤더이다.

```
float doIt (int bigAry[][12][5])
```

  - 예) 다음은 4차원 배열을 형식 매개변수로 갖는 함수의 헤더이다.
    - void func1(int a[][3][4][8]);
  - 예) 다음의 함수 헤더는 컴파일 에러이다.
    - int func2(int a[][][]);

이 곳에 배열 크기를 써줘야 한다.

Passing an array to a function

- 다음은 배열 원소의 값을 2배 하여 저장하는 **multiply()**함수를 사용한 프로그램이다.

```
 1  /* Read from keyboard & print integers multiplied by 2.
 2        Written by:
 3        Date:
 4  */
 5  #include <stdio.h>
 6  #define SIZE   5
 7
 8  //  Function Declarations
 9  void multiply (int* pAry, int size);
10
11  int main (void)
12  {
13  // Local Declarations
14     int  ary [SIZE];
15     int* pLast;
16     int* pWalk;
17
18  // Statements
```

multiply()함수의 선언부
➔ 배열을 인자로 받는다.

Passing an array to a function

```
19    pLast = ary + SIZE - 1;
20    for (pWalk = ary; pWalk <= pLast; pWalk++)
21       {
22        printf("Please enter an integer: ");
23        scanf ("%d", pWalk);
24       } // for
25
26    multiply (ary, SIZE);
27
28    printf ("Doubled value is: \n");
29    for (pWalk = ary; pWalk <= pLast; pWalk++)
30         printf (" %3d", *pWalk);
31
32    return 0;
33 }  // main
34
```

pLast ➔ 배열의 마지막 요소를 참조한다.

배열 초기화 ➔ scanf()를 통해 입력 받는다.

multiply() 호출

출력

Passing an array to a function

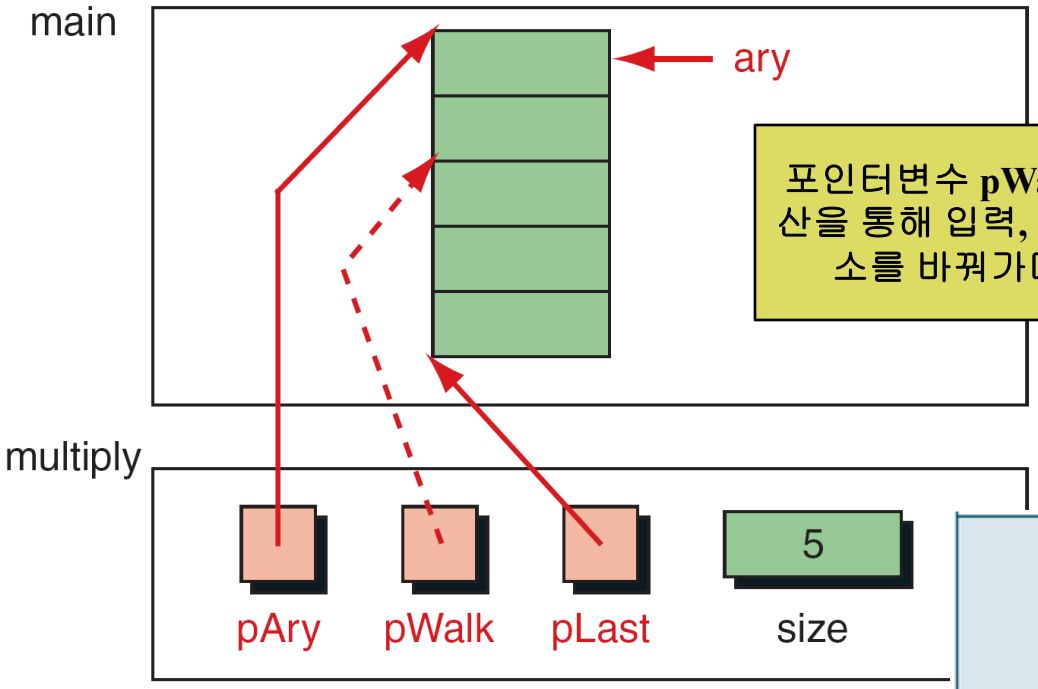
```
35  /* ==================== multiply ====================
36     Multiply elements in an array by 2
37        Pre   array has been filled
38              size indicates number of elements in array
39        Post  values in array doubled
40  */
41  void multiply (int* pAry, int size)
42  {
43  // Local Declarations
44     int* pWalk;
45     int* pLast;
46
47  // Statements
48     pLast = pAry + size - 1;
49     for (pWalk = pAry; pWalk <= pLast; pWalk++)
50         *pWalk = *pWalk * 2;
51     return;
52  }  // multiply
53  // ================ End of Program ================
```

배열과 배열크기를 인수로 받는다.

배열 원소 값 * 2

Passing an array to a function

- 다음은 **multiply()** 함수를 사용한 프로그램의 구조이다.



포인터변수 **pWalk**가 포인터연산을 통해 입력, 출력, 계산될 원소를 바꿔가며 참조한다.

실행 결과

```
Results:
Please enter an integer: 1
Please enter an integer: 2
Please enter an integer: 3
Please enter an integer: 4
Please enter an integer: 5
Doubled value is:
   2   4   6   8  10
```