

08-1. 배열의 선언과 사용

메모리에 저장 공간을 확보할 때 변수를 선언하는 과정에서 지금까지는 하나씩 지정하여 정해 주곤 했는데, 같은 형태의 많은 데이터를 반복문으로 처리하기 때문에 메모리에 연속적으로 저장해놓고 쪼개서 사용하는 방법을 사용한다. 같은 형태의 데이터면 ‘배열’로 묶어서 처리하면 편하다.

1) 배열의 선언

형태 : `int ary[5];` # 자료형 배열명[요소개수]

따로 배정해줄 때,

a 5바이트
b 5바이트
c 5바이트
d 5바이트
e 5바이트

배열로 묶어줄 때,

ary[0]	ary[1]	ary[2]	ary[3]	ary[4]
20 바이트				

- * 배열은 메모리에 연속된 공간이 할당되며, 하나의 이름을 사용한다.
- * 배열의 나누어진 조각을 ‘배열 요소’라고 한다.
- * 배열 요소는 배열명에 첨자를 붙여 표현하며 첨자는 0부터 시작한다.
- * 배열의 첨자는 (최대 배열 요소 개수 - 1)까지 사용한다.

- 예시

```

int main(void)
{
    int ary[5];

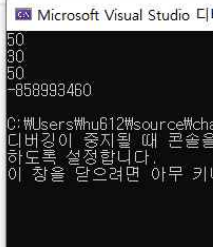
    ary[0] = 10;
    ary[1] = 20;
    ary[2] = ary[0] + ary[1];

    scanf("%d", &ary[3]);

    printf("%d\n", ary[2]);
    printf("%d\n", ary[3]);
    printf("%d\n", ary[4]);

    return 0;
}

```



scanf로 값을 입력, 50을 입력하면

10	20	30	50	
ary[0]	ary[1]	ary[2]	ary[3]	ary[4]

다음과 같이 배열에 할당할 수 있다.

ary[4]에는 값을 저장하지 않았으므로 쓰레기 값을 출력한다.

[참고] 배열의 첨자가 사용 범위를 넘어가서 사용하면 용도에 따라 결과가 달라진다.

예를 들어 ary[4]까지 지정했는데 ary[5]를 만들어내면 이 경우, 컴파일러가 경고 메시지로 알려주기도 하지만 배열 요소에 포인터 연산을 통해 접근하므로 확실한 에러 메시지를 표시하지는 않는다. 실행 단계에서 문제를 일으키게 되면 발견할 수 있으므로 버그를 찾아내기 쉽지 않다. 따라서 사용 범위를 벗어나지 않도록 주의해야 한다.

2) 배열 초기화

배열도 변수처럼 최소 할당된 저장 공간에 쓰레기 값이 저장되어 있다.

따라서 배열도 원하는 값을 가지려면 선언과 동시에 초기화를 해야한다.

배열을 초기화 하는 방법은 다음과 같다.

형태 : int ary1[5] = {1,2,3,4,5};

1	2	3	4	5
ary1[0]	ary1[1]	ary1[2]	ary1[3]	ary1[4]

초기화 시, 중괄호 안의 개수가 배열의 개수보다 작으면 남은 배열은 모두 0으로 채운다.

int ary3[5] = {1,2,3};

1	2	3	0	0
ary3[0]	ary3[1]	ary3[2]	ary3[3]	ary3[4]

int ary2[] = {1,2,3};

1	2	3
ary2[0]	ary2[1]	ary2[2]

int형 뿐 아니고 double형 배열과 char형 배열도 마찬가지로 비슷하게 선언하면 된다.

```
double ary4[5] = {1.0, 2.1, 3.0, 4.5, 5.4};
```

```
char ary5[5] = {'a', 'p', 'p', 'l', 'e'};
```

배열의 초기화는 선언 시 최소 한 번만 가능하다. 이후에는 배열 요소에 값을 일일이 대입해야 하며 지금처럼 중괄호를 사용한 대입 연산으로 한 번에 값을 바꾸는 것은 불가능하다.

3) 배열과 반복문

배열은 연속된 저장 공간을 할당하고 초기화할 수 있어서 유형의 변수가 많이 필요할 때 사용한다. 따라서 배열 요소를 일일이 변수처럼 떼어서 사용하기보다는, 반복문으로 사용한다.

-예시

```
int score[5];
```

```
int i;
```

```
int total = 0;
```

```
double avg;
```

```
for (i = 0; i < 5; i++)  
{  
    scanf("%d", &score[i]);  
}
```

```
for (i = 0; i < 5; i++)  
{  
    total += score[i];  
}
```

```
avg = total / 5.0;
```

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

```
printf("평균 : %.1lf\n", avg);
```



```
C:\Users\Whu612\source\chapte  
80 74 77 89 100  
80 74 77 89 100  
평균 : 84.0
```

4) sizeof 연산자를 활용한 배열 처리

배열은 보통 많은 양의 데이터를 처리할 때 쓰기 때문에 반복문 사용이 필수이다.

따라서 배열 요소의 개수가 바뀌면 배열을 처리하는 반복문을 모두 수정해야 하는 부담이 있다. 이 경우 배열 요소의 개수를 직접 계산해 반복문에 사용하는 방법이 있다.

형태 : $\text{sizeof}(\text{배열명}) / \text{sizeof}(\text{배열 요소})$

- 예시

```
count = sizeof(score) / sizeof(score[0]);

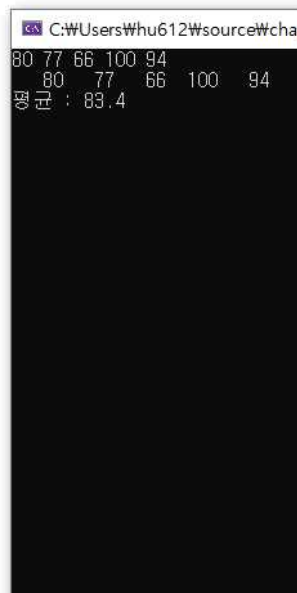
for (i = 0; i < count; i++)
{
    scanf("%d", &score[i]);
}

for (i = 0; i < count; i++)
{
    total += score[i];
}

avg = total / (double)count;

for (i = 0; i < count; i++)
{
    printf("%5d", score[i]);
}
printf("\n");

printf("평균 : %.1f\n", avg);
```



count = sizeof(score) / sizeof(score[0])
 => 5 = 20바이트 / 4바이트

5) 마무리 정리

- 배열을 선언하면 많은 변수를 한 번에 선언하는 효과를 볼 수 있다.
- 배열을 초기화 할 때는 중괄호를 사용한다.
- 배열은 주로 반복문으로 처리한다.
- 배열 전체 크기를 구할 때 sizeof 연산자를 사용한다.

구분	사용 예	기능
배열 선언	int ary[5]	int형 변수 5개를 한 번에 확보한다.
요소 사용	ary[0], ary[1], ..	배열 요소를 사용할 때는 첨자를 0부터 시작해 (요소 개수 - 1) 까지 사용한다.
초기화	int ary[5] = {1,2,3,4,5};	초기화는 중괄호 안에 값을 나열한다.

6) 확인 문제

1. 다음 설명에 따라 배열을 선언해보자.

- ⦿ 정수 5개를 저장할 배열
- ⦿ 실수 10개를 저장할 배열
- ⦿ 배열 요소 개수가 3개인 int형 배열
- ⦿ 첨자가 최대값이 4인 char형 배열

2. 다음 그림과 일치하도록 배열을 선언하고 초기화해보자.(int형)

1	2	3	0	0	0
---	---	---	---	---	---

3. 다음과 같이 초기화된 A 배열의 값을 복사해 B 배열을 채운 후 출력하는 프로그램을 작성해보자.

1	2	3	1	2	3	1	2	3	1
---	---	---	---	---	---	---	---	---	---

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

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

```
int B[10];
```

```
int i;
```

```
    
```

```
{
```

```
    
```

```
}
```

```
    
```

```
{
```

```
    
```

```
}
```

```
return 0;
```

```
}
```

정답)

1. 다음 설명에 따라 배열을 선언해보자.

⊙ 정수 5개를 저장할 배열

`int ary[5];`

⊙ 실수 10개를 저장할 배열

`double ary1[10];`

⊙ 배열 요소 개수가 3개인 int형 배열

`int ary2[3];`

⊙ 첨자가 최대값이 4인 char형 배열

`char ary3[5];`

2. 다음 그림과 일치하도록 배열을 선언하고 초기화해보자.(int형)

1	2	3	0	0	0
---	---	---	---	---	---

`int ary[6] = {1,2,3};`

3. 다음과 같이 초기화된 A 배열의 값을 복사해 B 배열을 채운 후 출력하는 프로그램을 작성해보자.

1	2	3	1	2	3	1	2	3	1
---	---	---	---	---	---	---	---	---	---

```
#include <stdio.h>

int main(void)
{
    int A[3] = {1,2,3};
    int B[10];
    int i;

    for (i=0; i<10; i++)
    {
        B[i] = A[i%3];
    }

    for (i=0; i<10; i++)
    {
        printf("%5d", B[i]);
    }
    return 0;
}
```

08-2. 문자를 저장하는 배열

평소 의미를 전달하기 위해 단어를 사용하는데, 단어는 알파벳을 연속으로 적고 그 순서에 따라 뜻이 달라진다. 이런 단어를 컴퓨터에서 데이터로 처리하기 위한 가장 좋은 방법은 배열이다. 따라서 char 배열로 문자를 저장해보자.

1) char형 배열의 선언과 초기화

char형 배열을 선언할 때는 저장할 문자열의 길이보다 최소한 하나 이상 크게 배열을 선언하는 것이다. 여분의 공간이 필요한 이유는 널 문자(\0)를 저장하기 위해서이다.

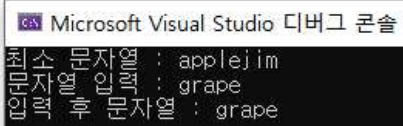
- 예시

```
// #pragma warning(disable:4996)
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    char str[80] = "applejim";

    printf("최소 문자열 : %s\n", str);
    printf("문자열 입력 : ");
    scanf("%s", str);
    printf("입력 후 문자열 : %s\n", str);

    return 0;
}
```



문자 상수로 하나씩 초기화 해도 되고 한번에 초기화해도 된다.

[널(빈) 문자의 용도]

초기화한 문자들은 배열의 처음부터 차례로 저장되어 문자열을 만드는데, 남은 배열 요소는 자동으로 0이 채워진다. 이렇게 char형 배열에 저장된 0을 '널 문자'라고 부른다.

모든 문자는 아스키 코드 값으로 저장되므로 결국 널 문자는 아스키 코드 값이 0인 문자를 말하며 문자 상수로는 \0 으로 표현된다.

a	p	p	l	e	j	i	m	\0	\0
---	---	---	---	---	---	---	---	----	----

널 문자는 문자열의 끝을 표시하는 용도로 쓰인다.

printf 이후 정확하게 문자열만 출력하는 이유도 널 문자가 있기 때문이다. printf 함수는 char 형 배열에서 널 문자가 나올때까지만 출력하도록 만들어졌다.

g	r	a	p	e	\0	\0	\0	\0	\0
---	---	---	---	---	----	----	----	----	----

scanf 함수는 사용자가 문자열을 입력한 다음에 자동으로 널 문자를 추가해 문자열의 끝에

표시를 한다. 따라서 입력 후 문자열로 grape만 출력되었다.

[char형 배열 선언 시 주의할 점]

- * 배열의 크기는 최대한 넉넉하게 선언해야 한다.
- * 배열 요소의 개수는 최소한 문자열 길이 + 1 이어야 한다.

2) 문자열 대입

char형 배열이 문자열을 저장하는 변수의 역할을 하기 때문에 초기화 이후에도 얼마든지 새로운 문자열을 저장할 수 있다.

이때 문자열의 길이가 다를 수 있으므로 strcpy 함수를 이용한다.

strcpy 함수는 char형 배열에 새로운 문자열을 저장하는 함수로, 저장할 문자열의 길이를 파악하여 딱 그 길이만큼만 char형 배열에 복사한다. 당연히 문자열 끝에 널 문자도 자동으로 붙여준다.

- 예시

```
// #pragma warning(disable:4996)
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>

int main(void)
{
    char str1[80] = "cat";
    char str2[80];

    strcpy(str1, "tiger");
    strcpy(str2, str1);

    printf("%s, %s\n", str1, str2);

    return 0;
}
```



#include <string.h> : 문자열 관련 함수 원형을 모아놓은 헤더 파일

strcpy함수 : strcpy(저장할 배열명, 저장할 문자열);

이때 저장할 배열명에는 문자열 상수를 사용할 수 없다.

strcpy("lion", "tiger") 이런식으로 못쓴다!

3) 문자열 전용 입출력 함수 : gets, puts

키보드로 문자열을 입력하는 문제를 생각해보자. scanf 함수는 char형 배열에 문자열을 입력할 수 있으나 중간에 빈칸이 있는 경우 빈칸 전까지 입력한다. 빈칸을 포함한 새로운 문자열 입력 방식을 알아보자.

gets : 빈칸을 포함하여 한 줄 전체를 문자열로 입력한다.

- 예시

Microsoft Visual Studio 디버그 콘솔

```
문자열 입력 : my name is jihye...
입력된 문자열 :
my name is jihye...
```

```
gets( char형 배열 );
```

[puts 함수]

문자열 상수나 char형 배열의 배열명을 주면 문자열을 화면에 출력한다. printf 함수의 문자열 출력 기능과 같다. 단, 문자열 출력 후 자동 줄바꿈 차이가 있다.

```
// #pragma warning(disable:4996)
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
```

Microsoft Visual Studio 디버그 콘솔
OK做做做做做?◀ㄹㄴ

할당된 배열에 널 문자가 없으니 이어지는 메모리 영역까지 출력한다.

4) 마무리 정리

- char형 배열은 문자열을 저장하는 변수의 역할을 하는 것으로 문자열을 직접 초기화할 수 있다.
- char형 배열은 문자열을 저장할 때는 대입 연산자 대신 strcpy 함수를 사용한다.
- char형 배열에 문자열을 입출력 할 때는 scanf, gets, puts, printf 등의 함수를 사용한다.

구분	사용 예	기능
char형 배열 초기화	<code>char str[30] = 'apple';</code>	char형 배열은 문자열로 초기화한다. 문자열의 끝에는 널 문자가 있다.
문자열 대입	<code>char str[80]; strcpy(str, 'apple');</code>	문자열 대입은 strcpy 함수를 사용한다. str 배열에 문자열 apple 저장.
문자열 입출력	<code>char str[80]; scanf("%s", str); gets(str) printf("%s", str); puts(str);</code>	scanf 함수는 하나의 단어만 입력 가능하다. gets 함수는 한 줄로 입력 가능하다. puts 함수로 문자열 출력 후 줄을 바꾼다.

5) 확인 문제

1. 다음 중 char형 배열이 초기화된 것은 동그라미, 아님 엑스 하시오.

- ⊙ char str[80] = {'p','i','g'};
- ⊙ char str[] = "elephant";
- ⊙ char str[5] = "apple";
- ⊙ char str[2] = {"sun", "moon"};

2. 다음 중 널 문자의 상수 표현법으로 맞는 것은?

NULL, /0, '0', '\0'

3. 다음 코드는 2개의 문자열을 입력 받아 위치를 바꾼 후 출력한다. 빈칸을 완성하시오.

```
#include <stdio.h>
#include <string.h>

int main(void)
{
    char str1[80], str2[80];
    char temp[80];

    printf("두 문자열 입력 : ");
    scanf("%s %s", str1, str2);
    printf("바꾸기 전 : %s, %s\n", str1, str2);

    strcpy( [ ] , [ ] );

    strcpy( [ ] , [ ] );

    strcpy( [ ] , [ ] );
    printf("바꾼 후 : %s, %s\n", str1, str2);
}
```

정답)

2. 다음 중 char형 배열이 초기화된 것은 동그라미, 아님 엑스 하시오.

- ⊙ char str[80] = {'p','i','g'}; ○
- ⊙ char str[] = "elephant"; ○
- ⊙ char str[5] = "apple"; ○
- ⊙ char str[2] = {"sun", "moon"}; x

2. 다음 중 널 문자의 상수 표현법으로 맞는 것은? '\0'

NULL, /0, '0', '\0'

3. 다음 코드는 2개의 문자열을 입력 받아 위치를 바꾼 후 출력한다. 빈칸을 완성하시오.

```
#include <stdio.h>
#include <string.h>

int main(void)
{
    char str1[80], str2[80];
    char temp[80];

    printf("두 문자열 입력 : ");
    scanf("%s %s", str1, str2);
    printf("바꾸기 전 : %s, %s\n", str1, str2);

    strcpy( temp , str1 ); #str1 문자열을 temp에 복사
    strcpy( str1 , str2 ); #str2 문자열을 str1에 복사
    strcpy( str2 , temp ); #temp 문자열을 str1에 복사
    printf("바꾼 후 : %s, %s\n", str1, str2);
}
```