

8장 배열



❖ 배열의 선언

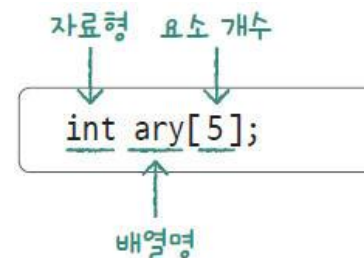
5명의 나이를 저장할 배열을 선언하고 사용하는 방법

소스 코드 예제8-1.c

```

01 #include <stdio.h>
02
03 int main(void)
04 {
05     int ary[5];           // int형 요소 5개의 배열 선언
06                           // ary는 array의 축약어
07     ary[0] = 10;          // 첫 번째 배열 요소에 10 대입
08     ary[1] = 20;          // 두 번째 배열 요소에 20 대입
09     ary[2] = ary[0] + ary[1]; // 첫 번째와 두 번째 요소를 더해 세 번째 요소에 저장
10     scanf("%d", &ary[3]); // 키보드로 입력받아 네 번째 요소에 저장
11
12     printf("%d\n", ary[2]); // 세 번째 배열 요소 출력
13     printf("%d\n", ary[3]);
14     printf("%d\n", ary[4]); // 마지막 배열 요소는 쓰레기 값
15
16     return 0;
17 }

```



실행결과

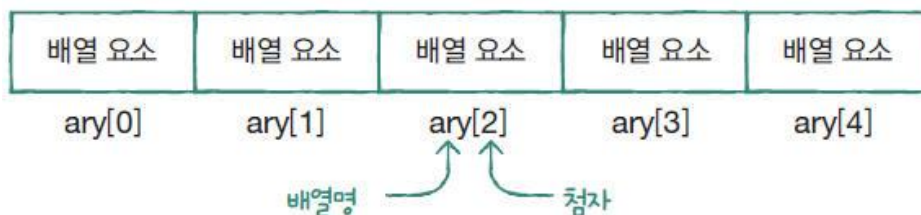
```

50
30
50
-858993460

```

❖ 배열 요소의 사용

- 배열 요소는 배열명과 첨자(index)로 하나의 변수처럼 사용한다.



```
ary[0] = 10;           // 7행. 첫 번째 배열 요소에 10 대입
ary[1] = 20;           // 8행. 두 번째 배열 요소에 20 대입
ary[2] = ary[0] + ary[1]; // 9행. 첫 번째와 두 번째 요소를 더해 세 번째 요소에 저장
```

- 첨자는 0부터 시작하며 최대 첨자는 요소의 수 - 1이다.

배열을 선언할 때

```
int ary[5];
```

배열 요소를 사용할 때

```
ary[0] ~ ary[4]
```

❖ 배열 초기화 (1/2)

- 중괄호 안에 초기값을 나열하면 앞에서부터 차례로 초기화된다.

```
int ary1[5] = {1,2,3,4,5};
```

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

- 요소의 수보다 초기값이 적으면 남은 요소는 0으로 초기화된다.

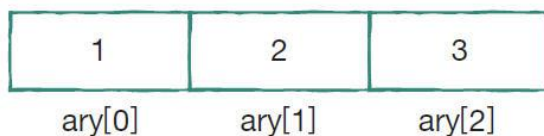
```
int ary2[5] = {1,2,3};
```

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

❖ 배열 초기화 (2/2)

- 초기화하면 배열 요소의 수를 생략할 수 있다.

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



- 각 자료형에 맞는 값으로 초기화한다.

```
double ary4[5] = {1.0, 2.1, 3.2, 4.3, 5.4};
```

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

❖ 배열과 반복문

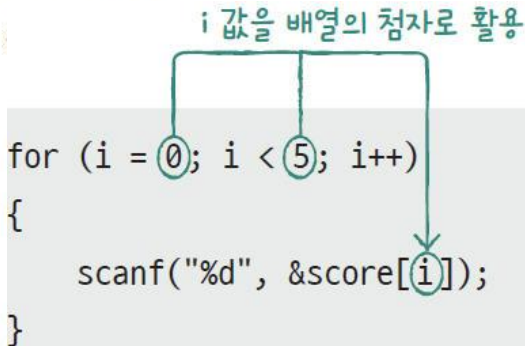
배열과 반복문을 사용한 성적 처리 프로그램

소스 코드 예제8-2.c

```

01 #include <stdio.h>
02
03 int main(void)
04 {
05     int score[5];
06     int i;
07     int total = 0;
08     i 값을 배열의 첨자로 활용
09
10     for (i = 0; i < 5; i++)
11     {
12         scanf("%d", &score[i]);
13     }
14

```



```

15     for (i = 0; i < 5; i++)
16     {
17         total += score[i];
18     }
19     avg = total / 5.0;
20
21     for (i = 0; i < 5; i++)
22     {
23         printf("%5d", score[i]);
24     }
25     printf("\n");
26
27     printf("평균 : %.1lf\n", avg);
28
29     return 0;
30 }

```

실행결과

80	95	77	84	100
80	95	77	84	100
평균 : 87.2				

❖ sizeof 연산자를 활용한 배열 처리

sizeof 연산자를 사용한 배열

소스 코드 예제8-3.c

```

01 #include <stdio.h>
02
03 int main(void)
04 {
05     int score[5];
06     int i;
07     int total = 0;
08     double avg;
09     int count;
10
11     count = sizeof(score) / sizeof(score[0]);
12
13     for (i = 0; i < count; i++)
14     {
15         scanf("%d", &score[i]);
16     }

```

```

18     for (i = 0; i < count; i++)
19     {
20         total += score[i];
21     }
22     avg = total / (double)count;
23
24     for (i = 0; i < count; i++)
25     {
26         printf("%5d", score[i]);
27     }
28     printf("\n");
29
30     printf("평균 : %.1lf\n", avg);
31
32     return 0;
33 }

```

실행결과

80	95	77	84	100
80	95	77	84	100
평균 : 87.2				

키워드로 끝내는 핵심 포인트

- ❖ 배열을 선언하면 많은 변수를 한 번에 선언하는 효과가 있다.
- ❖ 배열을 초기화할 때는 중괄호 { }를 사용한다.
- ❖ 배열은 주로 반복문으로 처리한다.
- ❖ 배열 전체의 크기를 구할 때 sizeof 연산자를 사용한다.

표로 정리하는 핵심 포인트

표 8-1 배열의 선언과 요소의 사용

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

❖ char형 배열의 선언과 초기화 (1/3)

문자열을 저장하는 char형 배열

소스 코드 예제8-4.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     char str[80] = "applejam";
06
07     printf("최초 문자열 : %s\n", str);
08     printf("문자열 입력 : ");
09     scanf("%s", str);
10     printf("입력 후 문자열 : %s\n", str);
11
12     return 0;
13 }
```

실행결과

최초 문자열 : applejam
문자열 입력 : grape
입력 후 문자열 : grape

// 문자열 초기화

// 초기화 문자열 출력

// 새로운 문자열 입력

// 입력된 문자열 출력

❖ char형 배열의 선언과 초기화 (2/3)

- 두 가지 초기화 방법

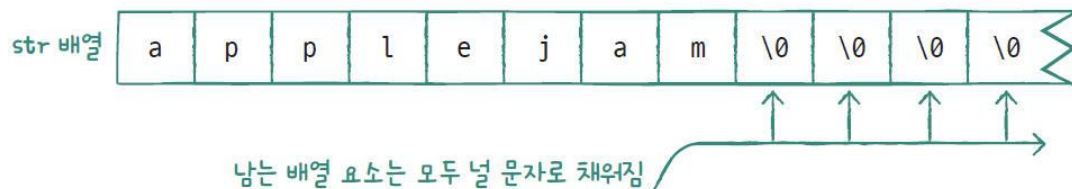
문자 상수로 하나씩 초기화

```
char str[80] = {'a', 'p', 'p', 'l', 'e', 'j', 'a', 'm'};
```

문자열 상수로 한 번에 초기화

```
char str[80] = "applejam";
```

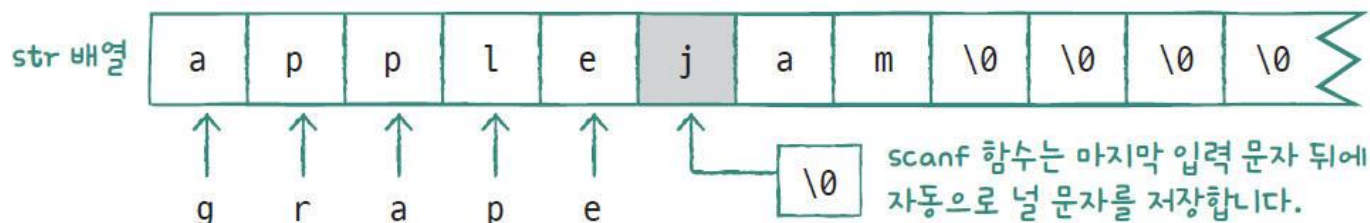
- 널문자를 저장할 공간이 있어야 한다.



❖ char형 배열의 선언과 초기화 (3/3)

- scanf 함수로 입력할 때는 배열명만 사용한다.

```
scanf("%s", str);
```



- 초기화하지 않는 경우 반드시 널문자도 저장해야 한다.

```
char str[80];    // 배열 선언, 초기화하지 않음
str[0] = 'a';    // 배열 요소에 직접 문자 대입
str[1] = 'p';
str[2] = 'p';
```

```
str[3] = 'l';
str[4] = 'e';
str[5] = '\0';
// 마지막 문자 다음에 반드시 널 문자 대입!
```

❖ 문자열 대입

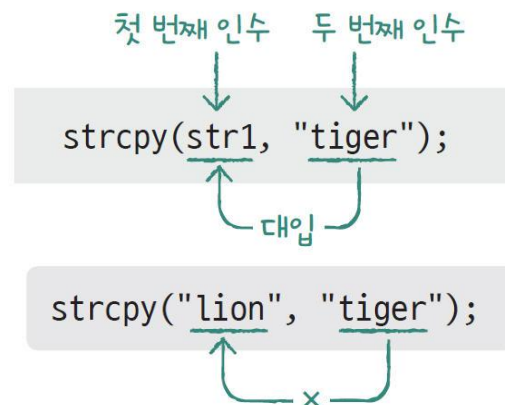
문자열을 대입하는 strcpy 함수

소스 코드 예제8-5.c

```

01 #include <stdio.h>
02 #include <string.h>           // 문자열 관련 함수 원형을 모아놓은 헤더 파일
03
04 int main(void)
05 {
06     char str1[80] = "cat";
07     char str2[80];
08
09     strcpy(str1, "tiger"); // str1 배열에 "tiger" 복사
10     strcpy(str2, str1);   // str2 배열에 str1 배열의 문자열 복사
11     printf("%s, %s\n", str1, str2);
12
13     return 0;
14 }

```



실행결과	
tiger, tiger	

❖ 문자열 전용 입출력 함수 : gets, puts

빈칸을 포함한 문자열 입력

소스 코드 예제8-6.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     char str[80];
06
07     printf("문자열 입력 : ");    // 입력 안내 메시지 출력
08     gets(str);                  // 빈칸을 포함한 문자열 입력
09     puts("입력된 문자열 : ");    // 문자열 상수 출력
10     puts(str);                  // 배열에 저장된 문자열 출력
11
12     return 0;
13 }
```

실행결과

문자열 입력 : Love is belief... 

입력된 문자열 :

Love is belief...

❖ 문자열 끝에 널문자가 없다면?

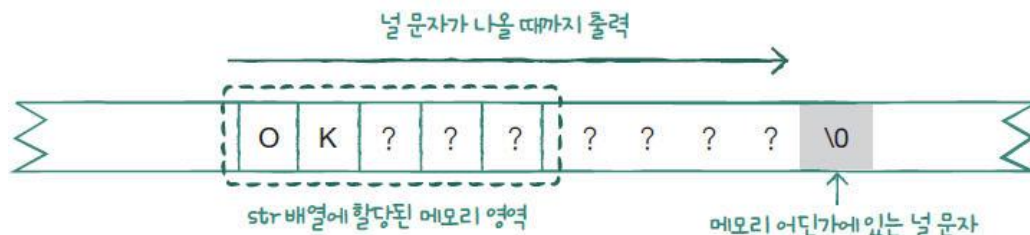
널 문자가 없는 문자열

소스 코드 예제8-7.c

```

01 #include <stdio.h>
02
03 int main(void)
04 {
05     char str[5];
06
07     str[0] = 'O';
08     str[1] = 'K';
09     printf("%s\n", str);
10
11     return 0;
12 }

```



실행결과

OK ㄹㄹㄹㄹㄹ? ㄹㄹㄹ*

키워드로 끝내는 핵심 포인트

- ❖ **char형 배열**은 문자열을 저장하는 변수 역할을 한다.
- ❖ char형 배열은 **문자열로 직접 초기화**할 수 있다.
- ❖ char형 배열에 문자열을 저장할 때는 **strcpy 함수**를 사용한다.
- ❖ 문자열 입출력은 scanf, **gets**, printf, **puts** 등의 함수를 사용한다.

표로 정리하는 핵심 포인트

표 8-2 문자열 처리

구분	사용 예	기능
char형 배열 초기화	<code>char str[80] = "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 gets 함수는 한 줄 입력 printf 함수는 문자열 출력 puts 함수는 문자열 출력 후 줄 바꿈