3. 변수의 이해

1. 변수 선언

- 요리를 하기에 앞서 그릇을 준비하듯이 C 프로그램을 작성하려면 변수 선언을 먼저 수행
- 국그릇, 밥그릇, 반찬그릇 등 다양한 그릇이 있는 것처럼 변수의 종류도 다양
- 소수점이 없는 값과 소수점이 있는 값을 담는 변수를 선언

```
int a;
float b;
```

- 두 문장으로 다음 그림과 같이 새로운 변수(그릇) 2개를 생성
- 변수(그릇)에 각각 정수와 실수를 담을 수 있음

그림 3-6 정수형 변수와 실수형 변수의 개념

3. 변수의 이해

1. 변수 선언

- 변수를 선언하는 방식은 다양
- 만약 정수형 변수 a와 b를 선언하고 싶다면 다음과 같은 방식을 사용할 수 있음
- 즉 변수의 종류가 같을 때는 변수를 개별적으로 선언해도 되고 콤마(,)를 사용하여 연속해서 선언해도 됨

```
int a;
int b;
```

==

```
int a, b;
```

3. 변수의 이해

1. 변수 선언

- 정수형 변수 a, 실수형 변수 b, 정수형 변수 c, 실수형 변수 d를 선언하는 기본 방식

❶ 가능

```
int a;
float b;
int c;
float d;
```

==

❷ 가능

```
int a, c;
float b, d;
```

≠

❸ 불가능

```
int a, float b;
int c, float d;
```

여기서 잠깐 세미콜론을 사용한 줄 표현

- 한 줄에 하나의 데이터 형식만 선언할 수 있다고 했으나 엄밀하게 말하면 '한 줄'이 아니라 '한 문장'이라고 해야 옳음
- ❷는 올바른 형식이며 세미콜론(;)으로 구분된 것은 완전히 분리 된 문장으로 취급되므로 ❶과 ❷는 같은 의미

❶

```
int a;
float b;
int c;
float d;
```

❷

```
int a;  float b;
int c;  float d;
```

2. 변수에 값을 담는 방법

▪ 대입 연산자와 변수의 위치

- '대입 연산자(=)를 사용하면 오른쪽의 것이 왼쪽에 대입된다'는 규칙
- 대입 연산자(=)의 왼쪽에는 반드시 무엇을 담는 그릇인 변수만 온다는 것을 알 수 있음

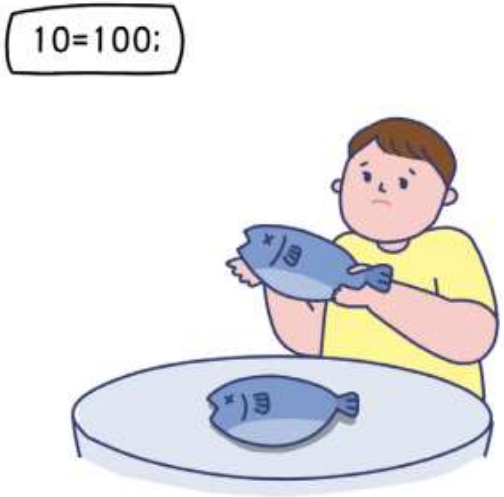

그림 3-15 값을 넣을 그릇이 없는 경우

3. 변수의 이해

2. 변수에 값을 담는 방법

▪ 대입 연산자와 변수의 위치

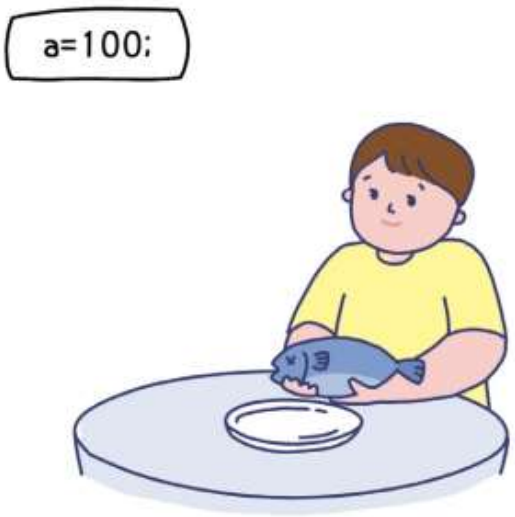

그림 3-16 값을 넣을 그릇이 있는 경우

- '대입 연산자(=)의 오른쪽에는 상수(숫자), 변수, 계산값이 모두 올 수 있다'는 규칙도 적용
- 결론적으로 대입 연산자의 왼쪽에는 변수만 올 수 있고 오른쪽에는 상수, 변수, 계산 값, 함수 등 무엇이든지 올 수 있음

4. 데이터 형식과 배열

3. 숫자형 데이터 형식

▪ 소수점이 없는 정수형

표 3-7 정수형 데이터 형식

데이터 형식	의미	크기	값의 범위
short	작은 정수형	2바이트	-2^{15}(-32768) ~ $2^{15}-1$(32767)
unsigned short	부호 없는 작은 정수형	2바이트	0 ~ $2^{16}-1$(65535)
int	정수형	4바이트	-2^{31}(약 -21억) ~ $2^{31}-1$(약 21억)
unsigned int	부호 없는 정수형	4바이트	0 ~ $2^{32}-1$(약 42억)
long int(또는 long)	큰 정수형	4바이트	-2^{31} ~ $2^{31}-1$
unsigned long	부호 없는 큰 정수형	4바이트	0 ~ $2^{32}-1$

- 정수형은 말 그대로 소수점이 없는 데이터를 입력하기 위해 사용하는 데이터 형식
- unsigned가 붙은 데이터 형식은 마이너스(-) 값이 없는 경우에 사용

4. 데이터 형식과 배열

3. 숫자형 데이터 형식

▪ 소수점이 있는 정수형

표 3-8 실수형 데이터 형식

데이터 형식	의미	크기	값의 범위
float	실수형	4바이트	약 $-3.4\times10^{38} \sim 3.4\times10^{38}$
double	큰 실수형	8바이트	약 $-1.79\times10^{308} \sim 1.79\times10^{308}$
long double	큰 실수형	8바이트	약 $-1.79\times10^{308} \sim 1.79\times10^{308}$

- float 형은 대개 소수 아래 일곱 자리까지의 정밀도를 나타내지만 double 형은 소 수점 아래 열여섯 자리 정도까지의 정밀도를 나타낼 수 있음

4. 문자형 데이터 형식

▪ 아스키코드

- 아스키코드는 컴퓨터에서 표현하는 문자(특히 키보드에 있는 영문자, 기호, 숫자 등)를 0~127에 대응한 코드

표 3-9 아스키코드

아스키코드	10진수	16진수
0 ~ 9	48 ~ 57	0x30 ~ 0x39
A ~ Z	65 ~ 90	0x41 ~ 0x5A
a ~ z	97 ~ 122	0x61 ~ 0x7A

- C에서는 숫자를 문자로도 표현

```
char ch = 'a';   ==   char ch = 97;
```

4. 문자형 데이터 형식

▪ 한 글자를 뜻하는 문자형

표 3-10 문자형 데이터 형식

데이터 형식	의미	크기	값의 범위
char	문자형 또는 정수형	1바이트	$-2^7(-128) \sim 2^7-1(127)$
unsigned char	문자형 또는 부호 없는 정수형	1바이트	$0 \sim 2^8-1(255)$

- char 형에는 문자뿐만 아니라 값의 범위에 해당하는 정수를 대입할 수 있음
- char 형을 1바이트 크기의 정수형으로 취급해도 상관없음
- char 형의 크기는 1바이트(8비트)이므로 표현할 수 있는 글자 수는 256가지이며 값의 범위는 –128~127
- 아스키코드의 0~127을 담을 수 있음

1. 배열의 이해

1. 배열을 사용하는 이유

▪ 배열의 개념

- 여러 개의 변수를 나란히 연결하는 개념
- 박스(변수)를 한 줄로 붙이고, 박스의 이름(aa)을 지정
- 각각의 박스는 aa[0], aa[1], ... 과 같이 첨자를 붙임

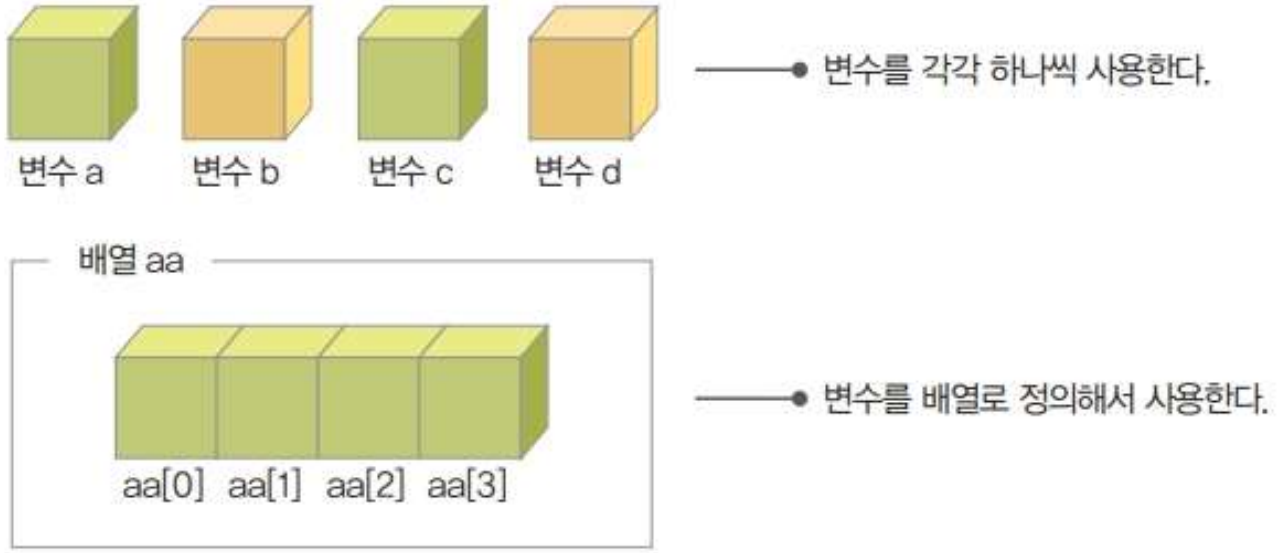

그림 8-1 배열의 개념

1. 배열의 이해

1. 배열을 사용하는 이유

▪ 배열의 선언 방법

```
데이터_형식  배열_이름[개수];
```

- 변수 4개를 담은 정수형 배열을 선언의 예

```
int aa[4];
```

- 배열을 사용하지 않는다면 각각의 변수를 int a, b, c, d;와 같이 선언해야 함
- 배열의 경우에는 첨자를 사용하여 aa[0], aa[1], aa[2], aa[3]과 같이 선언
- 배열을 4개 선언할 때는 첨자를 1~4가 아닌 0~3을 사용

구분	변수	배열
선언 예	int a, b, c, d;	int aa[4];
사용할 수 있는 변수 형식 예	a, b, c, d	aa[0], aa[1], aa[2], aa[3]

1. 배열의 이해

2. 배열의 활용 범위

- 배열의 첨자가 순서대로 변할 수 있도록 반복문과 함께 활용해야만 배열의 효율성이 극대화

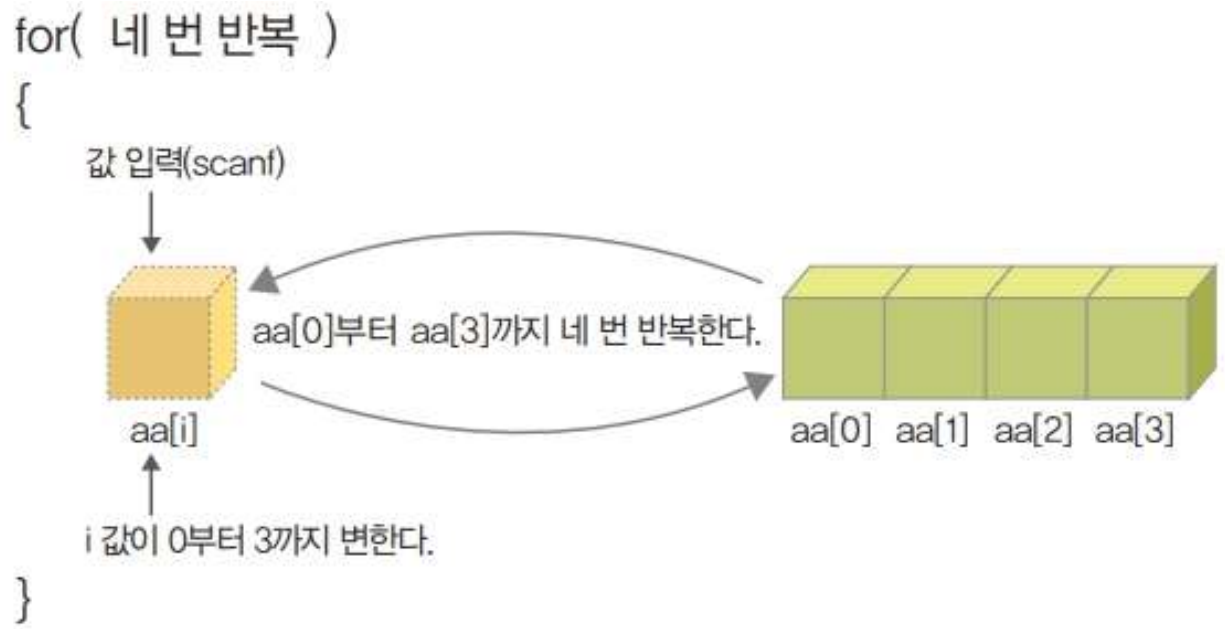

그림 8-2 for문을 활용한 배열값 입력

- for문을 네 번 돌면서 aa[i]의 첨자가 aa[0]~aa[3]으로 변하게 하면 변수 4개에 자동으로 값이 입력