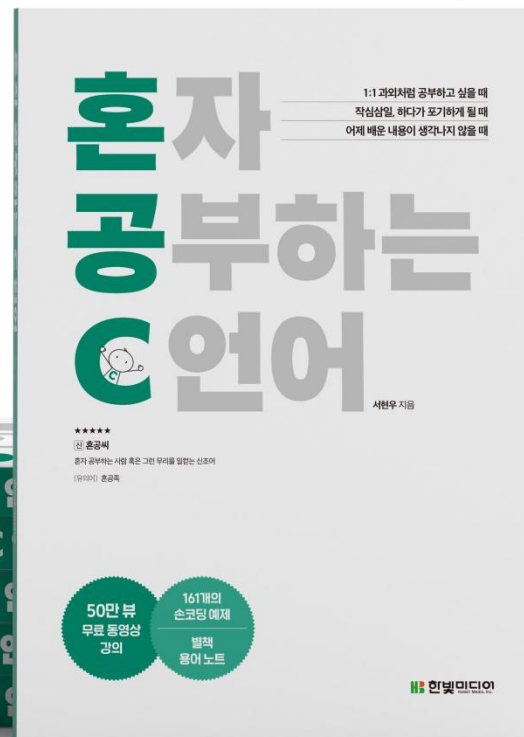


3장 변수와 데이터 입력



❖ 변수 선언 방법 (1/3)

- 자료형과 변수명으로 선언

변수의 선언과 사용

소스 코드 예제3-1.c

01 #include <stdio.h>

02

03 int main(void)

04 {

4행의 { 부터 23행의 }까지가
하나의 코드 블록이다.

05 int a; // int형 변수 a 선언

06 int b, c; // 2개의 int형 변수 b,c를 동시에 선언

07 double da; // double형 변수 da 선언

08 char ch; // char형 변수 ch 선언

09

int형 변수 선언

int a;

자료형

변수명

❖ 변수 선언 방법 (2/3)

```
10     a = 10;           // int형 변수 a에 정수 10 대입
11     b = a;           // int형 변수 b에 변수 a의 값 대입
12     c = a + 20;       // int형 변수 c에 변수 a의 값과 정수 20을 더한 값 대입
13     da = 3.5;        // double형 변수 da에 실수 3.5 대입
14     ch = 'A';        // char형 변수 ch에 문자 'A' 대입
15
16     printf("변수 a의 값 : %d\n", a);
17     printf("변수 b의 값 : %d\n", b);
18     printf("변수 c의 값 : %d\n", c);
19     printf("변수 da의 값 : %.1lf\n", da);
20     printf("변수 ch의 값 : %c\n", ch);
21
22     return 0;
23 }
```

 실행결과 ×

변수 a의 값 : 10
변수 b의 값 : 10
변수 c의 값 : 30
변수 da의 값 : 3.5
변수 ch의 값 : A

❖ 변수 선언 방법 (3/3)

- 대입 연산자(=)는 왼쪽 변수에 오른쪽 값 저장

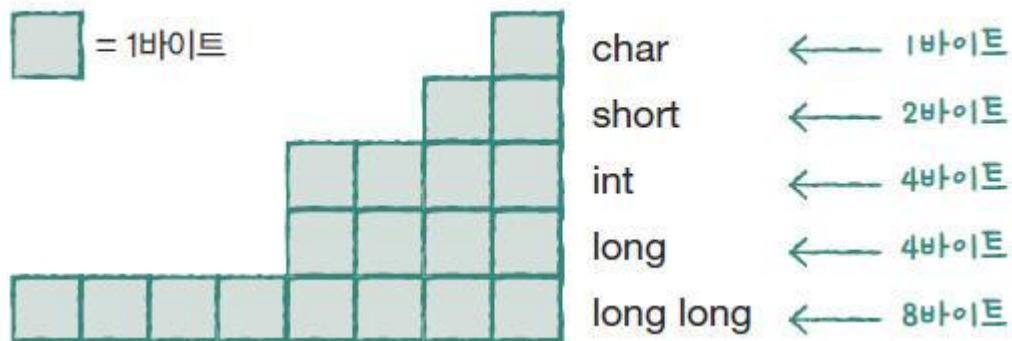
변수	상수, 변수, 수식	
a	= 10;	← 변수에 상수 대입
b	= a;	← 변수에 변수 대입
c	= a + 20;	← 변수에 수식 대입

- 대입 연산자 왼쪽 변수는 저장 공간, 오른쪽은 값이다.

```
int a, b;  
① 저장 공간 → a = 10;  
b = a; ← ② 값
```

❖ 정수 자료형

- 정수를 저장하는 자료형(data type)의 종류



- 자료형에 따라 저장할 수 있는 값의 범위 계산

$$-2^{\text{비트 수}-1} \sim 2^{\text{비트 수}-1}-1$$

❖ char형 변수

char형 변수의 사용

소스 코드 예제3-2.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     char ch1 = 'A';           // 문자로 초기화, 저장된 값은 문자의 아스키 코드 값
06     char ch2 = 65;           // 문자 'A'의 아스키 코드 값에 해당하는 정수로 초기화
07
08     printf("문자 %c의 아스키 코드 값 : %d\n", ch1, ch1);
09     printf("아스키 코드 값이 %d인 문자 : %c\n", ch2, ch2);
10
11     return 0;
12 }
```

실행결과

문자 A의 아스키 코드 값 : 65
아스키 코드 값이 65인 문자 : A

❖ 여러 가지 정수형 변수

여러 가지 정수형 변수

소스 코드 예제3-3.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     short sh = 32767;           // short형의 최댓값 초기화
06     int in = 2147483647;        // int형의 최댓값 초기화
07     long ln = 2147483647;       // long형의 최댓값 초기화
08     long long lln = 123451234512345; // 아주 큰 값 초기화
09
10     printf("short형 변수 출력 : %d\n", sh);
11     printf("int형 변수 출력 : %d\n", in);
12     printf("long형 변수 출력 : %ld\n", ln);
13     printf("long long형 변수 출력 : %lld\n", lln);
14         // long long형은 lld로 출력
15     return 0;
16 }
```

실행결과

```
short형 변수 출력 : 32767
int형 변수 출력 : 2147483647
long형 변수 출력 : 2147483647
long long형 변수 출력 : 123451234512345
```


❖ unsigned 정수 자료형 (1/2)

unsigned를 잘못 사용한 경우

소스 코드 예제3-4.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     unsigned int a;
06
07     a = 4294967295;    // 큰 양수 저장
08     printf("%d\n", a); // %d로 출력
09     a = -1;           // 음수 저장
10     printf("%u\n", a); // %u로 출력
11
12     return 0;
13 }
```

실행결과	
-1	
4294967295	

❖ unsigned 정수 자료형 (2/2)

- unsigned 자료형은 항상 양수만 저장하고 %u로 출력한다.

```

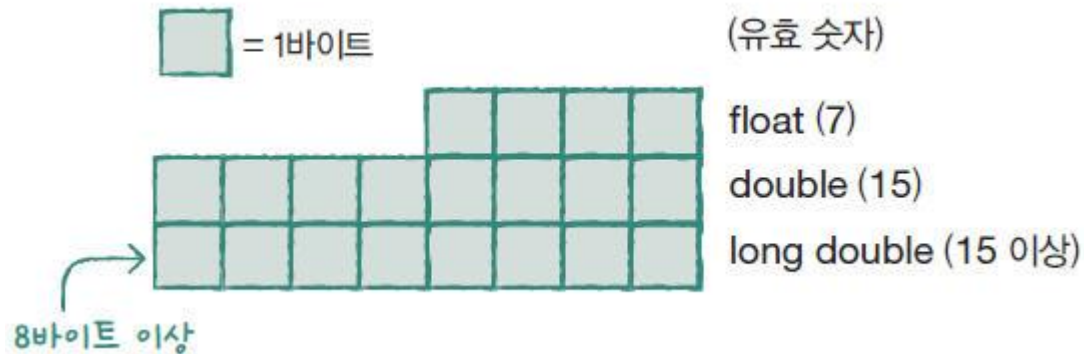
a = 4294967295;           // 7행. 큰 양수 저장
printf("%d\n", a);        // 8행. %d로 출력
a = -1;                   // 9행. 음수 저장
printf("%u\n", a);        // 10행. %u로 출력

```



❖ 실수 자료형 (1/2)

- 실수를 저장하는 자료형의 종류



- 유효 숫자가 많을 수록 더 정확한 값을 표현할 수 있다.

❖ 실수 자료형 (2/2)

유효 숫자 확인

소스 코드 예제3-5.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     float ft = 1.234567890123456789;           // 유효 숫자가 많은 값으로 초기화
06     double db = 1.234567890123456789;
07
08     printf("float형 변수의 값 : %.20f\n", ft);    // 소수점 이하 20자리까지 출력
09     printf("double형 변수의 값 : %.20lf\n", db);
10
11     return 0;
12 }
```

실행결과

```
float형 변수의 값 : 1.23456788063049316406
double형 변수의 값 : 1.23456789012345669043
```

❖ 문자열 저장 (1/3)

- 문자열은 char 배열에 저장

```
char fruit[6] = "apple" ;
```

↑ ↑
배열명 문자열의 길이+1 이상

- 배열의 크기는 `\0`(**널 문자** null character)를 위해 1이상 크게 확보

❖ 문자열 저장 (2/3)

char 배열에 문자열 저장

소스 코드 예제3-6.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     char fruit[20] = "strawberry";    // char 배열 선언과 문자열 초기화
06
07     printf("딸기 : %s\n", fruit);      // 배열명으로 저장된 문자열 출력
08     printf("딸기잼 : %s %s\n", fruit, "jam"); // 문자열 상수를 직접 %s로 출력
09
10     return 0;
11 }
```

실행결과

```
딸기 : strawberry
딸기잼 : strawberry jam
```

❖ 문자열 저장 (3/3)

- 새로운 문자열 저장은 strcpy 함수 사용

char 배열에 문자열 복사 [소스 코드](#) 예제3-7.c

```
01 #include <stdio.h>
02 #include <string.h>           // 문자열을 다룰 수 있는 string.h 헤더 파일 포함
03
04 int main(void)
05 {
06     char fruit[20] = "strawberry"; // strawberry로 초기화
07
08     printf("%s\n", fruit);         // strawberry 출력
09     strcpy(fruit, "banana");       // fruit에 banana 복사
10     printf("%s\n", fruit);         // banana 출력
11
12     return 0;
13 }
```

 실행결과 ×

```
strawberry
banana
```

❖ const를 사용한 변수

- const를 사용한 변수는 초깃값 수정 불가

const를 사용한 변수

소스 코드 예제3-8.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     int income = 0;           // 소득액 초기화
06     double tax;              // 세금
07     const double tax_rate = 0.12; // 세율 초기화
08
09     income = 456;             // 소득액 저장
10     tax = income * tax_rate;  // 세금 계산
11     printf("세금은 : %.11f입니다.\n", tax);
12
13     return 0;
14 }
```

실행결과

세금은 : 54.7입니다.

❖ 예약어와 식별자

- 예약어는 컴파일러와 약속된 단어, 식별자는 사용자가 만든 단어
- 식별자를 만드는 규칙
 - 알파벳 A~Z, a~z, 숫자 0~9, _(밑줄)만 사용
 - 숫자로 시작할 수 없다.
 - 대문자와 소문자는 서로 다른 식별자

식별자로 사용 가능한 예

```
apple  
make_money  
num7  
PrintName
```

식별자로 사용 불가능한 예

3times	←	숫자로 시작
non-stop	←	중간에 하이픈(-) 사용
Be_happy~	←	마지막에 틸드(~) 사용
apple tree	←	중간에 빈칸
short	←	예약어

키워드로 끝내는 핵심 포인트

- ❖ **변수 선언**으로 메모리에 저장 공간을 확보하며
대입 연산자(=)로 변수값을 초기화하거나 값을 저장한다.
초기화하지 않은 변수에는 **쓰레기 값**이 들어 있다.
- ❖ 변수의 형태를 **자료형**이라 하며 정수형과 실수형으로 나눈다.
- ❖ 변수에 **const**를 사용하면 상수처럼 사용할 수 있다.
- ❖ **예약어**는 컴파일러와 약속된 단어이고,
식별자는 사용자가 만들어낸 단어이다.

표로 정리하는 핵심 포인트 (1/2)

표 3-2 정수형의 종류

자료형	크기(Byte)	값의 저장 범위	출력 변환 문자
char	1	-128 ~ 127	%c 또는 %d
short	2	-32768 ~ 32767	%d
int	4	-2147483648 ~ 2147483647	%d
long	4	-2147483648 ~ 2147483647	%ld
long long	8	$-2^{63} \sim 2^{63}-1$	%lld
unsigned char	1	0 ~ 255	%u
unsigned short	2	0 ~ 65535	%u
unsigned int	4	0 ~ 4294967295	%u
unsigned long	4	0 ~ 4294967295	%lu
unsigned long long	8	$0 \sim 2^{64}-1$	%llu

표로 정리하는 핵심 포인트 (2/2)

표 3-3 실수형의 종류

자료형	크기(Byte)	값의 저장 범위	유효 숫자	출력 변환 문자
float	4	$-3.4 \times 10^{38} \sim 3.4 \times 10^{38}$	7	%f
double	8	$-1.79 \times 10^{308} \sim 1.79 \times 10^{308}$	15	%lf
long double	8 이상	double형과 같거나 큰 범위	15 이상	%Lf

표 3-4 예약어

구분	예약어
자료형	char double enum float int long short signed struct union unsigned void
제어문	break case continue default do else for goto if return switch while
기억클래스	auto extern register static
기타	const sizeof typedef volatile

❖ scanf 함수의 사용법 (1/2)

scanf 함수를 사용한 키보드 입력

소스 코드 예제3-9.c

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

```
02
```

```
03 int main(void)
```

```
04 {
```

```
05     int a;
```

```
06
```

```
07     scanf("%d", &a);    // 여기서 변수 a에 값 입력
```

```
08     printf("입력된 값 : %d\n", a); // 입력한 값 출력
```

```
09
```

```
10     return 0;
```

```
11 }
```

변수명

scanf("%d", &a);

int형 변환 문자 변수명 앞에 붙인다

실행결과

10

입력된 값 : 10

계속하려면 아무 키나 누르세요...

❖ scanf 함수의 사용법 (2/2)

scanf 함수를 사용한 연속 입력

소스 코드 예제3-10.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     int age;                // 나이는 정수형
06     double height;          // 키는 실수형
07
08     printf("나리와 키를 입력하세요 : "); // 입력 안내 메시지 출력
09     scanf("%d%lf", &age, &height);        // 나리와 키를 함께 입력
10     printf("나리는 %d살, 키는 %.1lfcm입니다\n", age, height); // 입력값 출력
11
12     return 0;
13 }
```

실행결과


나리와 키를 입력하세요 : 17 187.5
나리는 17살, 키는 187.5cm입니다


❖ 문자와 문자열의 입력


문자와 문자열 입력

소스 코드 예제3-11.c

```
01 #include <stdio.h>
02
03 int main(void)
04 {
05     char grade;           // 학점을 입력할 변수
06     char name[20];        // 이름을 입력할 배열
07
08     printf("학점 입력 : ");
09     scanf("%c", &grade);  // grade 변수에 학점 문자 입력
10     printf("이름 입력 : ");
11     scanf("%s", name);    // name 배열에 이름 문자열 입력, &를 사용하지 않는다.
12     printf("%s의 학점은 %c입니다.\n", name, grade);
13
14     return 0;
15 }
```

 실행결과 ✕

학점 입력 : A 

이름 입력 : 홍길동 

홍길동의 학점은 A입니다.

키워드로 끝내는 핵심 포인트

- ❖ **scanf 함수**로 입력할 때 변수 앞에 &를 사용한다.
- ❖ **둘 이상의 값을 입력**할 때는 `Space Bar`, `Tab`, `Enter` 로 구분한다.
- ❖ **문자열 입력**은 char 배열을 이용하며 &기호를 사용하지 않는다.

표로 정리하는 핵심 포인트

표 3-5 자료형에 따른 입력 변환 문자

데이터 종류	자료형	크기(Byte)	입력 변환 문자
정수	(unsigned) short	2	%hd (%hu)
	(unsigned) int	4	%d (%u)
	(unsigned) long	4	%ld (%lu)
	(unsigned) long long	8	%lld (%llu)
실수	float	4	%f
	double	8	%lf
	long double	8, 10, 12, 16	%Lf
문자	char	1	%c
문자열	char 배열	가변적	%s