자료형과 변수, 자료표현 방법

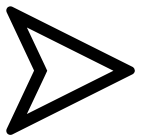
제주대학교 컴퓨터공학과
변영철 교수
(ycb@jejunu.ac.kr)

자료와 정보

- 어떤 값, 가령 나이 20은 자료(data)
- 70.5라는 몸무게 숫자도 자료
- "홍길동"이라는 이름 문자열도 자료
- 자료들이 모여서 의미가 생기면 정보(Information)
  - 3개의 자료, 홍길동, 20, 70.5이 모이면 '홍길동의 나이는 20이고, 몸무게는 70.5이다.'라는 의미(semantics)를 만들 수 있음.
  - 이것이 정보

자료의 종류

- 정수 (int)
- 실수 (double)
- 문자 (char)
- 문자열



자료 값은 상수,
이를 저장하기
위한 것이 변수

상수의 종류

상수: 항상 일정한 수

- 정수형 상수 (예, 20)
- 실수형 상수 (예, 70.5)
- 문자 상수 (예, 'A')
- 문자열 상수 (예, "홍길동")

표 2-3 상수 종류

종류	표현 방법	사용 예
정수	0~9, +, − 기호 사용	10, −5, +20, 0
실수	0~9, +, −, .(소수점) 기호 사용	3.4, −1.7, .5, 10.0
문자	문자를 작은 따옴표로 묶음	'A', 'b', '0', '*',
문자열	하나 이상의 문자를 큰 따옴표로 묶음	"A", "banana"

변수의 종류

변하는 수, 상수(값)를 저장하는 공간

- 정수형 변수 (예, int a;)
- 실수형 변수 (예, double b;)
- 문자형 변수 (예, char c;)
- 문자열 변수 → 문자 배열이나 포인터로 구현

변수의 크기

```
printf("%d, %d\n", sizeof(int), sizeof(float));
printf("%d, %d\n", sizeof(double), sizeof(char));
```

자료형	크기(byte)
int	4
float	4
double	8
char	1
문자열(char 배열)	가변적

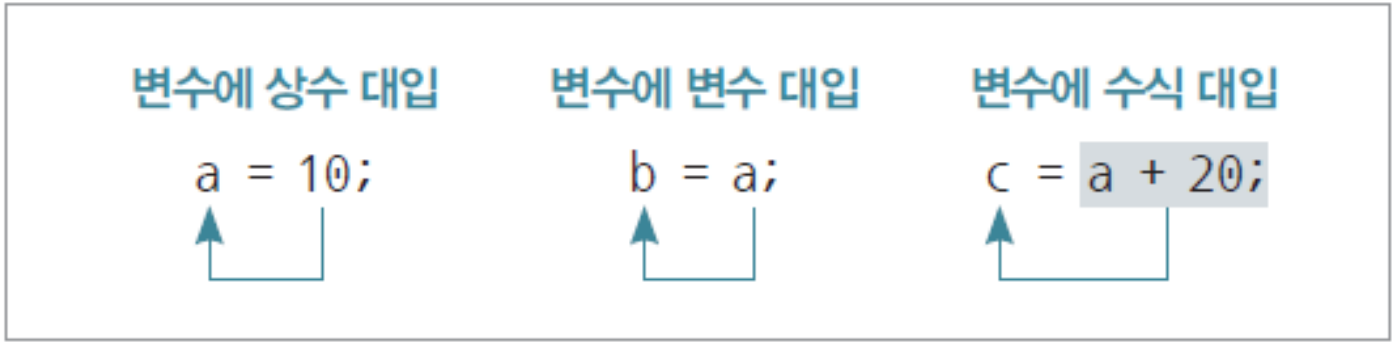
sizeof 연산자 → sizeof(자료형, 혹은 변수이름)

변수를 만드는 방법

- 변수 정의
- 정만선알 → 정의는 만드는 것(메모리에), 선언은 알리는 것(컴파일러에게)
- 변수 a를 정의하는 예

변수에 값 할당

- 대입 연산자(=)는 연산자 왼쪽 변수에 오른쪽 값 저장
  - 대입 연산자 왼쪽(left) 변수 → l-value
  - 대입 연산자 오른쪽(right) 값, 혹은 변수 → r-value



```c
#include <stdio.h>

int main(void)
{
    int a;
    int b, c;
    double da;
    char ch;

    a = 10;
    b = a;
    c = a + 20;
    da = 3.5;
    ch = 'A';

    printf("변수 a의 값 : %d\n", a);
    printf("변수 b의 값 : %d\n", b);
    printf("변수 c의 값 : %d\n", c);
    printf("변수 da의 값 : %.1lf\n", da);
    printf("변수 ch의 값 : %c\n", ch);

    return 0;
}
```

실행 결과

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

변수 초기화

예제 3-2 쓰레기값의 출력과 초기화 방법

```
1.  #include <stdio.h>
2.
3.  int main(void)
4.  {
5.      int a;                  // 9행에서 대입 연산으로 초기화
6.      int b = 20;             // 변수 선언과 동시에 초기화
7.      int c;                  // 초기화하지 않음
8.
9.      a = 10;                 // a에 10 대입, 초기화
10.     printf("a:%d\n", a);
11.     printf("b:%d\n", b);
12.     printf("c:%d\n", c);
13.
14.     return 0;
15. }
```

실행 결과

```
a:10
b:20
c:-858993460
```

변수 초기화

- 쓰레기(garbage) 값
  - 변수를 정의만 할 뿐 값을 할당하지 않으면 '이상한 쓸모 없는 쓰레기 값'이 들어가 있음.
- 변수 초기화
  - 따라서 변수를 만든 후에는 항상 초기화하는 것이 바람직함.

정수형 변수

- 정수가 들어가는 변수

자료형	크기(바이트)
short	2
int	4
long	4
long long	8

```
short a;
int b;
long c;
long long d;

printf("%d", sizeof(d));
```

정수형 변수

예제 3-4 여러 가지 정수형 변수

```
1. #include <stdio.h>
2.
3. int main(void)
4. {
5.     short sh = 32767;                          // short형의 최댓값 초기화
6.     int in = -2147483648;                      // int형의 최솟값 초기화
7.     long ln = 2147483647;                      // long형의 최댓값 초기화
8.     long long lln = 123451234512345;           // 아주 큰 값 초기화
9.
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);  // long long형은 lld로 출력
14.
15.    return 0;
16. }
```

실행 결과

```
short형 변수 출력 : 32767
int형 변수 출력 : -2147483648
long형 변수 출력 : 2147483647
long long형 변수 출력 : 123451234512345
```

정수형 변수

- short
  - 2 바이트 크기
  - 작은 정수를 저장할 경우 이용
- int
  - 정수형 변수는 일반적으로 int로 정의
  - 4바이트 크기

정수형 변수

- long (X)
  - 4 바이트
  - 옛날 16비트 컴퓨터에서는 int 자료형 크기가 2바이트
  - 따라서 옛날 컴퓨터에서 더 큰 변수를 만들려면 long을 이용했어야 했음.
- long long
  - int 크기의 두배 → 8 바이트
  - 아주 큰 정수를 다뤄야 할 경우 이용
  - 메모리 낭비 가능성

문자형 변수

- 컴퓨터에서 처리하는 모든 문자는 0~127 사이의 정수(아스키 코드)로 저장됨.
- 즉, 문자도 결국은 정수로 저장됨. 출력만 아스키 문자로 출력
- 저장된 아스키 코드(0~127)를 printf 에서 %c로 출력하면 화면에 해당 아스키 문자가 출력됨.
- %d로 출력하면 해당 십진수 숫자로 출력됨.

문자형 변수

예제 3-3 char형 변수의 사용

```
1.  #include <stdio.h>
2.
3.  int main(void)
4.  {
5.      char ch1 = 'A';      // 문자로 초기화
6.      char ch2 = 65;       // 문자 'A'의 아스키 코드값으로 초기화
7.
8.      printf("문자 %c의 아스키 코드값 : %d\n", ch1, ch1);
9.      printf("아스키 코드값이 %d인 문자 : %c\n", ch2, ch2);
10.
11.     return 0;
12. }
```

실행 결과

```
문자 A의 아스키 코드값 : 65
아스키 코드값이 65인 문자 : A
```

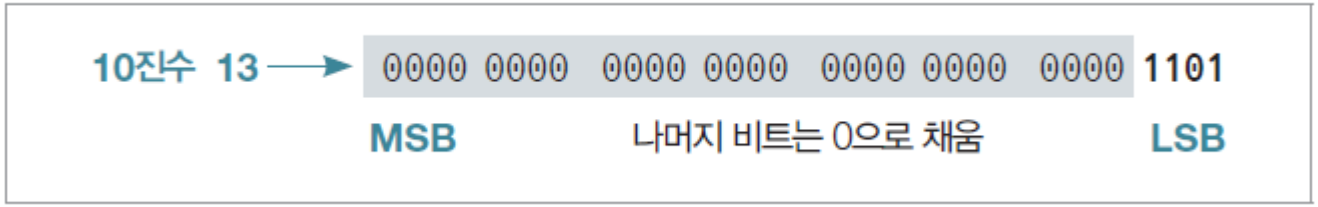
문자형 변수

- 아스키 코드 65번은 어떤 모양인지 출력해 보세요.
- 16진수 0x41의 아스키 코드는 어떤 모양인지 출력해 보세요.

정수값 표현 방법

int a = 13; //4바이트

"모든 데이터는 컴퓨터 안에서는 2진수로 표현된다."



MSB(Most Significant Bit) LSB(Least Significant Bit)

가장 왼쪽 비트:
0이면 양수,
1이면 음수

정수값 표현 방법

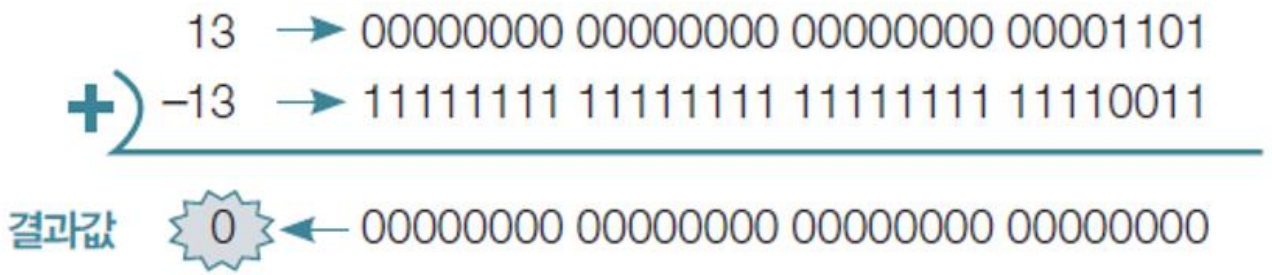
- int b = 13; //10진수로 초기화
- 원할 경우 2진수 혹은 16진수로 초기화 가능 (아래).
- 이진수로 초기화한 경우
  - int b = 0b00000000000000000000000000001101;
- 16진수로 초기화한 경우
  - int b = 0x0000000E;

정수값 표현 방법

int b = -13; //음수

-13 = 13의 2의 보수로 표현

- 절대값 13의 2진수 → 00000000 00000000 00000000 00001101
- 0과 1을 바꿈(1의 보수) → 11111111 11111111 11111111 11110010
- 1의 보수에 1을 더함 → 11111111 11111111 11111111 11110011



정수값 표현 방법

- 13을 8비트로 표현하면?

  0000 1101

- -13을 8비트로 표현하면? (※13의 2의 보수)

  0000 1101 => 1의 보수는? 1111 0010
  1111 0010 에 1을 더하면?

  (정답)

정수값 표현 방법

- 127을 8비트로 표현하면?

  0111 1111

- -127을 8비트로 표현하면?(※127의 2의 보수)

  0111 1111 => 1의 보수는? 1000 0000
  1000 0000 에 1을 더하면?

  (정답)

정수값 표현 방법

- 8비트로 정수(음수와 양수)를 표현할 경우 어떤 수들을 표현할 수 있을까?

0000 0000 (0)
0111 1111 (+127)
1000 0001 (-127)
1000 0000 (-128)
1111 1111 (-1)

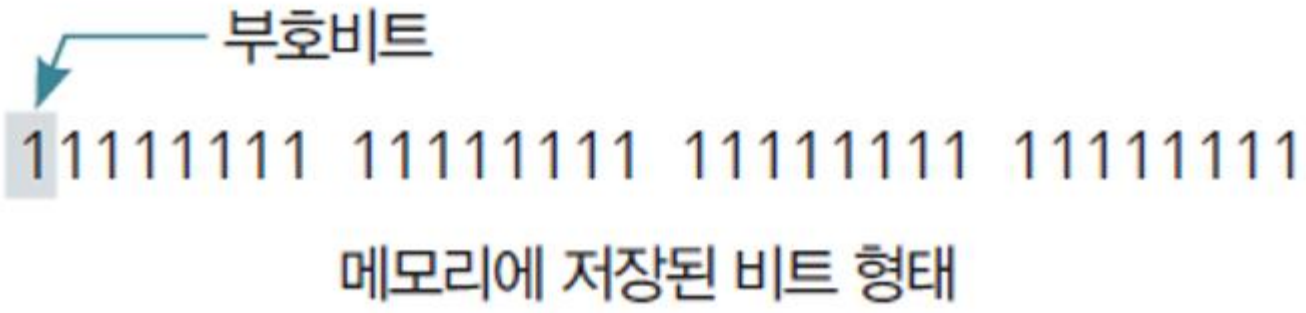
따라서 8비트는 -128에서 +127까지 수를 표현 가능

정수값 표현 방법

- 4비트로 표현할 수 있는 수의 범위는?

수 알아내기

- 아래 2진수는 10진수로 몇인가?



- 부호비트가 1이므로 음수이다.
- 2의 보수를 구한 후 –를 붙인다.

unsigned 정수형

- 정수는 양수와 음수 모두 표현하며, 가장 왼쪽 비트(MSB)는 부호(+, -) 표현
- 양수만 표현할 경우 2배 큰 수 표현가능

표 3-3 unsigned 정수형의 종류

자료형	크기(Byte)	값의 저장 범위	출력 변환문자
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

unsigned 정수형

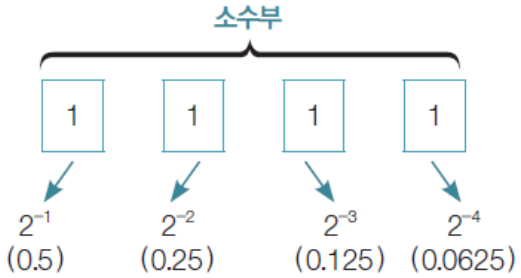
1. unsigned int 변수 a를 정의하자.
2. 변수 a의 모든 비트를 1로 할당하자.
3. 변수 a를 %u와 %d로 출력하자.

실수값 표현 방법

- 십진수로 표현 (우리가 흔히 쓰는 방식)
  - 12.5 (고정 소수점 방식)
  - $1.25 * 10^1 = 0.25 * 10^2$ (부동 소수점 방식)

실수값 표현 방법

- 이진수로 표현 (컴퓨터에서)
  - 12.5 => 1100.1 => $1.1001 * 2^3$



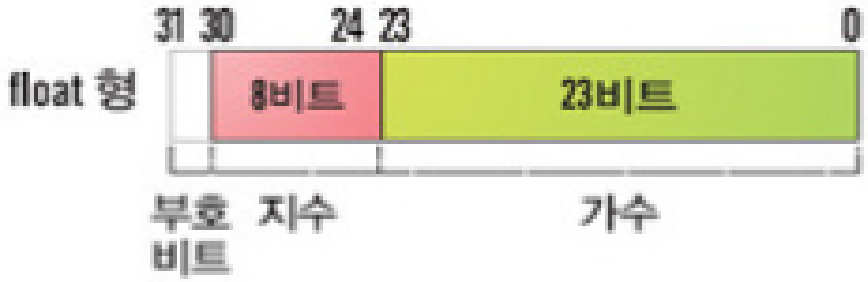
소수부 2진수	10진수 값 계산식	소수부 10진수
0 0 0 0	0	0
0 0 0 1	0.0625	0.0625
0 0 1 0	0.125	0.125
0 0 1 1	0.125 + 0.0625	0.1875
0 1 0 0	0.25	0.25
0 1 0 1	0.25 + 0.0625	0.3125
0 1 1 0	0.25 + 0.125	0.375
0 1 1 1	0.25 + 0.125 + 0.0625	0.4375
1 0 0 0	0.5	0.5
1 0 0 1	0.5 + 0.0625	0.5625
1 0 1 0	0.5 + 0.125	0.625
1 0 1 1	0.5 + 0.125 + 0.0625	0.6875
1 1 0 0	0.5 + 0.25	0.75
1 1 0 1	0.5 + 0.25 + 0.0625	0.8125
1 1 1 0	0.5 + 0.25 + 0.125	0.875
1 1 1 1	0.5 + 0.25 + 0.125 + 0.0625	0.9375

  - 1001 => 소수(가수) 부
  - 3 => 지수

실수값 표현 방법

12.5 => 1100.1 => 1.1001 * 2^3

float 자료형 (32비트) 표현 방식



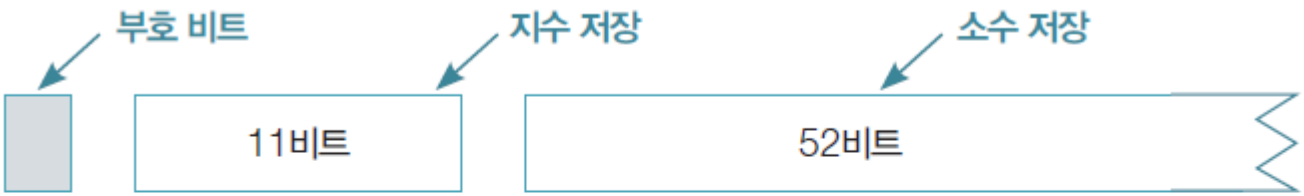
소수 (가수) 부분 : 1001
새로운 지수부분 : 지수(3) + 127 = 130 => 1000 0010

(정답) 0 10000010 10010000000000000000000

IEEE 754 표준(standard) 방식.

실수값 표현 방법

double 자료형 (64비트) 표현 방식



첫째, 가장 왼쪽의 비트는 부호비트며 양수는 0, 음수는 1로 표시합니다.

둘째, 부호비트 다음부터 11비트는 지수값을 의미합니다.

셋째, 나머지 52비트는 소수값을 의미합니다.

IEEE 754 표준(standard) 방식.

문자열 표현 방법

- 문자열은 여러 문자를 연결한 것
- 문자가 들어가는 변수 100개를 만들어 보자.
- 수많은 변수를 만든다?

  char a, b, c, d, e, f, … ;

- 배열 이용

  char a[100]; //a[0] ~ a[99]

문자열 표현 방법

- 문자 배열을 이용한 문자열 저장

예제 3-7 char 배열에 문자열 저장

```
1.  #include <stdio.h>
2.
3.  int main(void)
4.  {
5.      char fruit[20] = "strawberry";              // char 배열 선언과 초기화
6.
7.      printf("딸기 : %s\n", fruit);               // 배열명으로 strawberry 출력
8.      printf("딸기잼 : %s %s\n", fruit, "jam");   // 문자열은 %s로 출력
9.      printf("좋아하는 과일 : %s\n", fruit);
10.
11.     return 0
12. }
```

실행 결과

```
딸기 : strawberry
딸기잼 : strawberry jam
좋아하는 과일 : strawberry
```

문자열 표현 방법

- 문자열 복사는 strcpy 함수 이용

예제 3-8 char 배열에 문자열 복사

```
1.  #include <stdio.h>
2.  #include <string.h>                    // string.h 헤더 파일 포함
3.
4.  int main(void)
5.  {
6.      char fruit[20] = "strawberry";    // strawberry 초기화
7.
8.      printf("%s\n", fruit);            // strawberry 출력
9.      strcpy(fruit, "banana");         // fruit에 banana 복사
10.     printf("%s\n", fruit);           // banana 출력
11.
12.     return 0;
13. }
```

실행 결과

```
strawberry
banana
```

문자열 표현 방법

- 문자열 자료형은 없다.
- 문자열 변수는 없다.
- 문자 배열을 만든 후 문자열을 저장하는 것!

값을 바꿀 수 없는 변수

- const 키워드로

예제 3-9 const를 사용한 변수

```
1.  #include <stdio.h>
2.
3.  int main(void)
4.  {
5.      int income = 0;                     // 소득액 초기화
6.      double tax;                         // 세금
7.      const double tax_rate = 0.12;       // 세율 초기화
8.
9.      income = 456;                       // 소득액 저장
10.     tax = income * tax_rate;            // 세금 계산
11.     printf("세금은 : %.1lf입니다.\n", tax);
12.
13.     return 0;
14. }
```

실행 결과

```
세금은 : 54.7입니다.
```

값을 바꿀 수 없는 변수

- const 사용 시 반드시 변수 만들 때 초기화 해야 함.
- 이후 값을 바꿀 수 없음 (constant 니까.. 상수니까..)

예약어와 식별자

- 예약어(reserved word) → 이미 컴퓨터에서 사용하고 있는 word를 예약어라고 함

표 3-5 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

예약어와 식별자

- 식별자(identifier)는 식별되는 것들
- 우리가 원하는 대로 만드는 변수명, 함수명도 식별 가능하여 식별자라고 함
- 규칙에 맞는 식별자들
  - apple, num7, make_money, PrintName
- 잘못된 식별자들 (에러 발생)
  - 3times // 숫자로 시작
  - non-stop // 중간에 하이픈(-) 사용
  - Be_happy~ // 마지막에 틸드(~) 사용
  - apple tree // 중간에 빈 칸이 있음
  - short // 예약어

공부한 내용

- C언어 자료형을 이해한다.
- 상수와 변수를 이해한다.
- 변수 출력 방법을 안다.
- 자료 표현 방법을 안다.
- 상수형(const) 변수를 이해한다.
- 예약어와 식별자가 무엇인지 이해한다.