# 데이터의 표현

```
#include <stdio.h>

int main()
{
    printf("Data Representation in Computers\n");
    printf("Instructor: Keon Myung Lee\n");
    return 0;
}
```

# 내 용

- 데이터의 표현 방법
  - 수의 표현방법
    - 정수
    - 소수점있는 숫자(부동소수점 수)
  - 문자의 표현방법
    - 영문 : ASCII
    - 한글
  - 멀티미디어 데이터의 표현방법

- C 프로그래밍 언어 소개

# 데이터의 표현

- 프로그래밍에서 데이터의 종류
  - 수(number)
    - 정수 (integer)
    - 소수점있는 수
  - 문자 (character)
    - 영문자

# 데이터의 표현

- 컴퓨터의 데이터 표현
  - 2진수 기반 메모리의 데이터 저장 및 CPU 데이터 처리
  - 2진수에 의한 데이터 표현 요구
- 수(number)의 2진수 표현
  - 10진수를 2진수로 변환
  - 음수 표현
  - 소수점있는 수 표현
- 문자의 표현
  - 2진수로 표현

# 정수의 표현

- **진법** (number systems)
  - 10**진법** (decimal)
    - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 **사용**
    - 10**이 될 때 자리 올림**
  - 2**진법** (binary)
    - 0, 1 **사용**
    - 2**가 될 때 자리 올림**
  - 8**진법** (octal)
    - 0, 1, 2, 3, 4, 5, 6, 7 **사용**
    - 8**이 될 때 자리 올림**
  - 16**진법** (hexadecimal)
    - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F **사용**
    - 16**이 될 때 자리 올림**

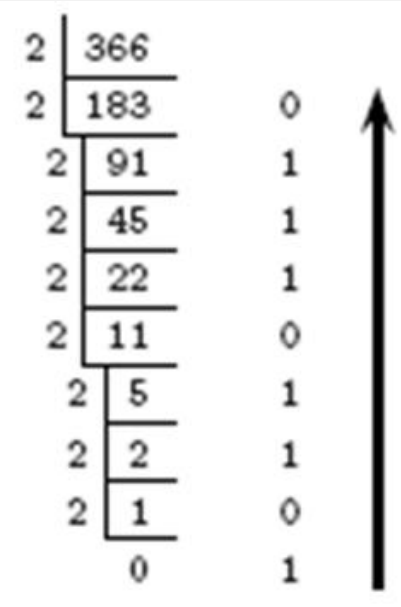
| Decimal | Binary | Octal | Hexadecimal |
|---|---|---|---|
| 0 | 0000 | 0 | 0 |
| 1 | 0001 | 1 | 1 |
| 2 | 0010 | 2 | 2 |
| 3 | 0011 | 3 | 3 |
| 4 | 0100 | 4 | 4 |
| 5 | 0101 | 5 | 5 |
| 6 | 0110 | 6 | 6 |
| 7 | 0111 | 7 | 7 |
| 8 | 1000 | 10 | 8 |
| 9 | 1001 | 11 | 9 |
| 10 | 1010 | 12 | A |
| 11 | 1011 | 13 | B |
| 12 | 1100 | 14 | C |
| 13 | 1101 | 15 | D |
| 14 | 1110 | 16 | E |
| 15 | 1111 | 17 | F |

http://base-conversions.blogspot.kr/2008/12/base-conversions-topic-that-students.html

http://base-conversions.blogspot.kr/2008/12/base-conversions-topic-that-students.html

# 정수의 표현

- 10진수 ⇒ 2진수, 8진수, 16진수




$366_{10}$
$= 101101110_2$

$= \underline{101101110}_2$
$= 556_8$

$= \underline{101101110}_2$
$= 16E_{16}$

Image source : http://introcs.cs.princeton.edu/java/51data/

# 정수의 표현

- 음수는 어떻게 표현할까?
  - 컴퓨터는 0 또는 1만 사용해야 함
  - 음수의 부호 (-)를 어떻게 표현할까?

- 코드화(encoding)
  - 어떤 내용을 직접 표현할 수 없거나 표현하기 싫을 때 다른 기호를 사용하여 표현
  - 기호로 표현된 것으로 부터 원래 내용을 복원해 내는 것 : 복호화 (decoding)
  - 컴퓨터에 음수 기호를 직접 나타낼 수 없기 때문에 코드화하여 표현
    - 부호절대값 표현
    - 2의 보수 표현

# 정수의 표현

- **부호절대값** (signed magnitude) **표현**
  - **맨 앞의** bit**를 부호를 나타내기 위해 사용**
    - 0 : **양수**
    - 1 : **음수**
  - **두 개의** 0**이 존재**
  - **양수와 양수간의 연산복잡**

    **예. 덧셈**
    - **절대값 크기 비교**
    - **절대값이 큰 것의 부호사용**
    - **절대값이 큰 것에서 작은 것 빼기**

| | |
|---|---|
| 3 | 011 |
| 2 | 010 |
| 1 | 001 |
| 0 | 000 |
| -0 | 100 |
| -1 | 101 |
| -2 | 110 |
| -3 | 111 |

# 정수의 표현

- 2의 보수 (2's complement) 표현
  - 맨 앞의 bit값이 0인 것으로 0부터 양수 표현
  - 맨 앞의 bit값이 1인 것은 절대값이 작아지는 순서로 -1부터 음수 표현
  - 한 개의 0만 존재
  - 덧셈 및 뺄셈 간단

예) 3 + (-2) = 1

```
  011      3
+110     -2
1001      1
```

2-3 = 2+(-3) = -1

```
  010      2
+101     -3
 111     -1
```

예) 3 + 2 = 5

```
  011      3
+010      2
 101     -3
```

표현가능범위 초과 (overflow)

| | |
|---|---|
| 3 | 011 |
| 2 | 010 |
| 1 | 001 |
| 0 | 000 |
| -1 | 111 |
| -2 | 110 |
| -3 | 101 |
| -4 | 100 |

0이상 양수

음수

3자리 이진수 2의 보수 표현

# 정수의 표현

- **오버플로우**(overflow, **범람**) **문제**
  - **메모리는 원고지의 칸처럼 정해진 공간에 2진수를 저장**
  - **나타낼 수 있는 수의 범위를 초과하면 잘못된 결과가 나타남**
  - **계산 중간에라도 표현 가능 범위를 초과하면, 최종 결과도 잘못됨**

**예.** 3bit**를 사용하는** 2**의 보수 표현 방법**

- **표현가능 범위** : -4 ~ 3

예) 3 + 2 = 5

```
 011      3
+010      2
 101     -3
```

| | |
|---|---|
| 3 | 011 |
| 2 | 010 |
| 1 | 001 |
| 0 | 000 |
| -1 | 111 |
| -2 | 110 |
| -3 | 101 |
| -4 | 100 |

# 정수의 표현

- 2의 보수 표현 방법에 따른 표현 가능 범위
  - 1 byte = 8 bits
    - **가능한 2진수 조합수** : 256 ($2^8$) $\Rightarrow$ {0, 1, …, $2^8$-1 = 255}
    - **가능한 범위** : –128 ~ 127 (–$2^7$ ~ $2^7$-1)
  - 2 bytes = 16 bits
    - **가능한 2진수 조합수** : 65,536 ($2^{16}$) $\Rightarrow$ {0, 1, …, $2^{16}$-1 = 65,535}
    - **가능한 범위** : –32,768 ~ 32,767
  - 4 byte = 32 bits
    - **가능한 2진수 조합수** : 4,294,967,296 ($2^{32}$)
    - **가능한 범위** : –2,147,483,648 ~ 2,147,483,647

# 정수의 표현

- **음수가 불필요한 정수 표현**
  - **나이, 연도, 수입 등**
  - **2의 보수 표현을 사용하게 되면 표현 가능 범위 중 반만 활용**

    **예**. 1byte **사용시** 2**의 보수 표현**
    - **–128 ~ 127 범위 사용**
    - –128~ –1 **범위는 사용하지 않으므로 낭비**

  - **음수를 사용하지 않는** (unsigned) **표현 방법 사용**

    **예**. 1byte **사용시** 2**의 보수 표현**
    - **0 ~ 255 범위 표현**

# 정수의 표현

- C 언어의 정수형 자료형(data type)
  - 저장 크기에 따른 정수를 나타내는 데이터의 종류 지정
    - 32bit 컴파일러의 경우

- **char** (character)
  - 1 byte
- **short** (short integer)
  - 2 bytes
- **int** (integer)
  - 4 bytes
- **long** (long integer)
  - 4 bytes

| Data type | Size | Min value | Max value |
|---|---|---|---|
| `char` | 1 | -128 | 127 |
| `unsigned char` | 1 | 0 | 255 |
| `short` | 2 | -32,768 | 32,767 |
| `unsigned short` | 2 | 0 | 65,535 |
| `int` | 4 | -2,147,483,648 | 2,147,483,647 |
| `unsigned int` | 4 | 0 | 4,294,967,295 |
| `long` | 4 | -2,147,483,648 | 2,147,483,647 |
| `unsigned long` | 4 | 0 | 4,294,967,295 |

# 소수점 있는 수의 표현

- **소수점 있는 수(실수)**
  - **소수점있는 수**

    **예**. 123.45  –45.67  45,345.12  $1.234 \times 10^{-3}$

  - **C언어에서 표현**
    - **쉼표(,) 사용 불가**
    - **일반표현**
      - 123.45  –45.67  45345.12  0.001234
    - **지수표현**
      - $-12.34 \times 10^{-4}$ ⇒ –12.34**e–4** **또는** –12.34**E–4**

# 소수점 있는 수의 표현

- **고정 소수점 수** (fixed point number)
  - **소수점의 위치를 고정시켜 놓은 표현 방법**

    **예**. 123.45

- **부동 소수점 수** (floating point number)
  - **소수점의 위치를 변경할 수 있는 표현 방법**

    **예**. $1.2345 = 12345 \times 10^{-4} = 123.45 \times 10^{-2}$

    **유효숫자 지수**
  - IEEE 754 standard
    - **C언어에서 사용하는 부동소수점 수 표현 표준**

- **어떤 표현 방법도 실제 실수의 값을 정확히 표현할 수 없다**
  - **원고지 칸과 같이 한정된 메모리 공간에 표시해야 하는 제약**

# 소수점 있는 수의 표현

❖ 소수점 있는 수의 2진수 표현 (고점소수점 수)

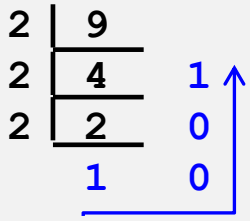
| 지수 | 3 | 2 | 1 | 0 | | -1 | -2 | -3 | -4 |
|---|---|---|---|---|---|---|---|---|---|
| 위치값 | $2^3$ | $2^2$ | $2^1$ | $2^0$ | | $2^{-1}$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ |
| 예 | 1 | 0 | 0 | 1 | . | 1 | 1 | 1 | 0 |

$1001.1110_2 = 1\times2^3 + 1\times2^0 + 1\times2^{-1} + 1\times2^{-2} + 1\times2^{-3}$

$= 1\times8 + 1\times1 + 1\times0.5 + 1\times0.25 + 1\times0.125$

$= 9.875$

$= 9 + 0.875$




$0.875 \times 2 = 1.750$

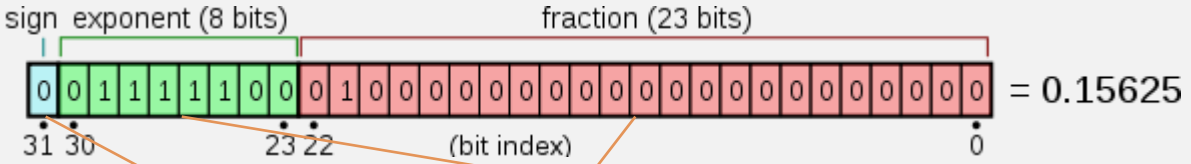
$0.750 \times 2 = 1.50$

$0.5 \times 2 = 1.0$

$0.0 \times 2 = 0.0$

# 부동소수점수의 표현

- IEEE 754 standard
  - **float** (single-precision floating point number **단정밀도 부동소수점수**)
    - 4 bytes **사용**
    - **부호**(sign) bit : 1 bit
    - **지수**(exponent) width : 8 bits
    - **가수**(significand, mantissa; **유효숫자**) precision : 23 bits
      - **의미적으로는 실제** 24bits **표현 효과**




$$= (-1)^{\text{sign}}(1.b_{-1}b_{-2}...b_{-23})_2 \times 2^{e-127}$$

$sign = 0$

$$1 + \sum_{i=1}^{23} b_{-i}2^{-i} = 1 + 2^{-2} = 1.25$$

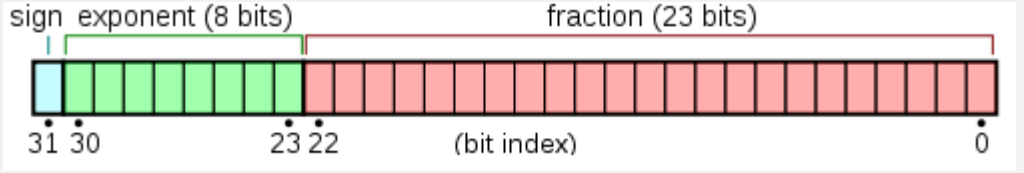
$$2^{(e-127)} = 2^{124-127} = 2^{-3}$$

$$value = 1.25 \times 2^{-3} = 0.15625$$

http://en.wikipedia.org/wiki/Single_precision

# 부동소수점수의 표현

- float **표현**




$$(1)_{10} = (1.0)_2 \times 2^0$$

$$= (-1)^{\text{sign}}(1.b_{-1}b_{-2}...b_{-23})_2 \times 2^{e-127}$$

- Sign : 0
- Exponent : 0, e = $127_{10}$ = 0111 $1111_2$
- Fraction (mantissa): 0 = 000 0000 0000 0000 0000 0000
- 0-01111111-00000000000000000000000 = $3f800000_H$

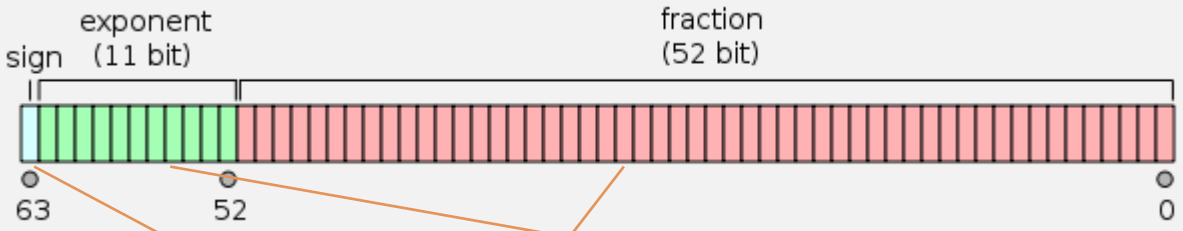
$$0.375 = (1.1)_2 \times 2^{-2}$$

- Sign : 0
- Exponent : -2, e = $125_{10}$ = 0111 $1101_2$
- Fraction (mantissa): 1 = 100 0000 0000 0000 0000 0000
- 0-01111101-10000000000000000000000 = $3ec00000_H$

http://en.wikipedia.org/wiki/Single_precision

# 부동소수점수의 표현

- **double** (double-precision floating point number **배정밀도 부동소수점수**)
  - 8 bytes **사용**
  - **부호**(sign) bit : 1 bit
  - **지수**(exponent) width : 11 bits
  - **가수**(significand, mantissa; **유효숫자**) precision : 52 bits
    - **의미적으로는 실제** 53bits **표현 효과**




$$= (-1)^{sign}(1.b_{-1}b_{-2}...b_{-52})_2 \times 2^{e-1023}$$

$$value = (-1)^{sign}(1 + \sum_{i=1}^{52} b_{-i}2^{-i}) \times 2^{(e-1023)}$$

# 부동소수점수의 표현

- double 형 표현 예
  - $0000\ 0000\ 0000\ 0000_{16} = 0$
  - $8000\ 0000\ 0000\ 0000_{16} = -0$

0000 0000 0000

1000 0000 0000

  - $7ff0\ 0000\ 0000\ 0000_{16} = \infty$
  - $fff0\ 0000\ 0000\ 0000_{16} = -\infty$

0111 1111 1111

1111 1111 1111

  - $0010\ 0000\ 0000\ 0000_{16} \approx 2.2250738585072014 \times 10^{-308}$
    - (Min normal positive double)
  - $7fef\ ffff\ ffff\ ffff_{16} \approx 1.7976931348623157 \times 10^{308}$
    - (Max double)

0000 0000 0001

0111 1111 1110

  - $3ff0\ 0000\ 0000\ 0000_{16} = 1$
  - $3fd5\ 5555\ 5555\ 5555_{16} \approx 1/3$

# 부동소수점수의 표현

- 표현 가능 범위
  - `float` 32 bits

    $1.18 \times 10^{-38} < |X| < 3.40 \times 10^{38}$

    underflow　　overflow
  - `double` 64bits

    $2.23 \times 10^{-308} < |X| < 1.79 \times 10^{308}$
  - `long double` 80bits

    $3.37 \times 10^{-4932} < |X| < 1.18 \times 10^{4932}$


http://www.tony.seriouslyinternet.com/CO2501/Resources/Floating%20point%20numbers.pdf

# 문자의 표현

- ASCII **코드** (American Standard Code for Information Interchange)
  - **영문자를 나타내는 코드**
  - 7bits**를 사용하여 정의하나**, 1byte**를 사용하여 저장**

| $b_7$ | $b_6$ | $b_5$ | | 0 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1 0 0 | 1 0 1 | 1 1 0 | 1 1 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bits | | | Column → | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| $b_4$ | $b_3$ | $b_2$ | $b_1$ | Row ↓ | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | NUL | DLE | SP | 0 | @ | P | ` | p |
| 0 | 0 | 0 | 1 | 1 | SOH | DC1 | ! | 1 | A | Q | a | q |
| 0 | 0 | 1 | 0 | 2 | STX | DC2 | " | 2 | B | R | b | r |
| 0 | 0 | 1 | 1 | 3 | ETX | DC3 | # | 3 | C | S | c | s |
| 0 | 1 | 0 | 0 | 4 | EOT | DC4 | $ | 4 | D | T | d | t |
| 0 | 1 | 0 | 1 | 5 | ENQ | NAK | % | 5 | E | U | e | u |
| 0 | 1 | 1 | 0 | 6 | ACK | SYN | & | 6 | F | V | f | v |
| 0 | 1 | 1 | 1 | 7 | BEL | ETB | ' | 7 | G | W | g | w |
| 1 | 0 | 0 | 0 | 8 | BS | CAN | ( | 8 | H | X | h | x |
| 1 | 0 | 0 | 1 | 9 | HT | EM | ) | 9 | I | Y | i | y |
| 1 | 0 | 1 | 0 | 10 | LF | SUB | * | : | J | Z | j | z |
| 1 | 0 | 1 | 1 | 11 | VT | ESC | + | ; | K | [ | k | { |
| 1 | 1 | 0 | 0 | 12 | FF | FC | , | < | L | \ | l | \| |
| 1 | 1 | 0 | 1 | 13 | CR | GS | - | = | M | ] | m | } |
| 1 | 1 | 1 | 0 | 14 | SO | RS | . | > | N | ^ | n | ~ |
| 1 | 1 | 1 | 1 | 15 | SI | US | / | ? | O | _ | o | DEL |

| | | |
|---|---|---|
| A | 01000001 | 65 |
| C | 01000011 | 67 |
| E | 01000101 | 69 |

| | | |
|---|---|---|
| a | 01100001 | 97 |
| c | 01100011 | 99 |
| e | 01100101 | 101 |

http://en.wikipedia.org/wiki/ASCII

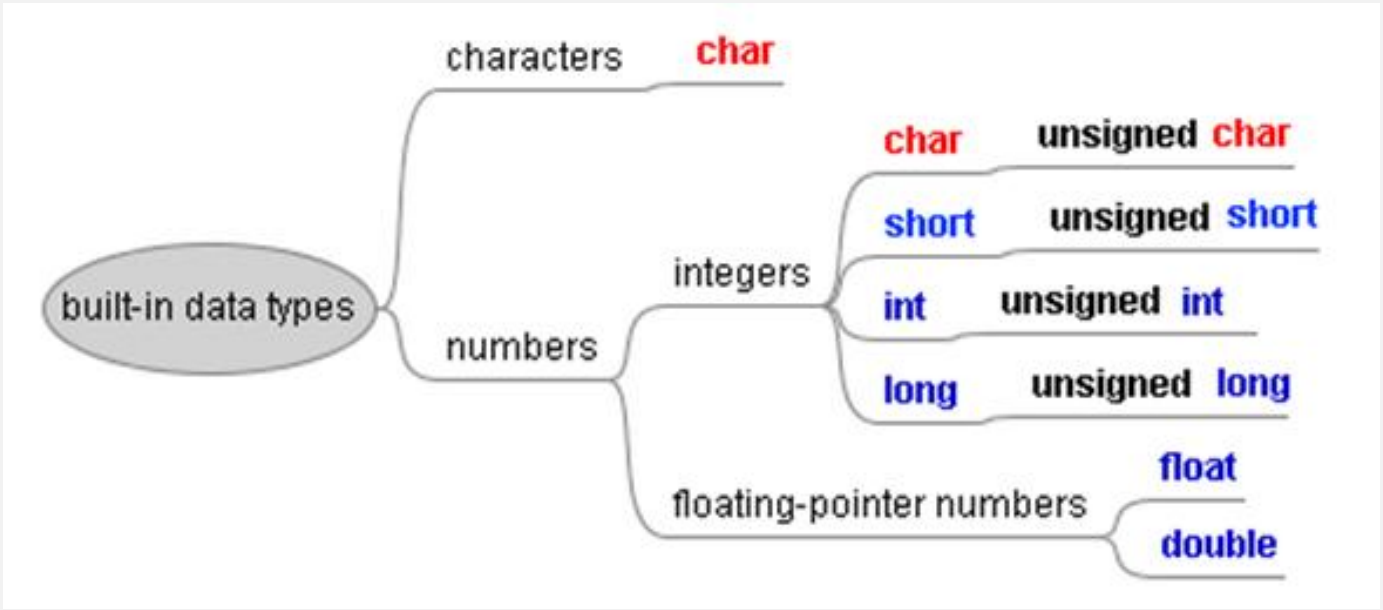
# 문자의 표현

- C 언어의 영문자형 자료형(data type)
  - char 형 (character)
    - 1 byte 사용
    - 문자 하나를 값으로 가질 수 있는 데이터 형
    - 문자 상수 : 단일 따옴표 안에 포함된 문자, 숫자, 특수기호 등

      'A'  '$'  'b'  '7'  'y'  '!'  'M'  'q'

# C언어의 자료 표현

- C언어의 자료형



# 문자의 표현

- 한글의 표현
  - 2 bytes 사용
  - 조합형 코드
    - 초성, 중성, 종성 코드로 한글 조합


  - 완성형 코드
    - 한글을 순서대로 나열 후 코드 부여


http://news.korean.go.kr/online/see/hangulstory/hangulstory.jsp?idx=29#

# 문자의 표현

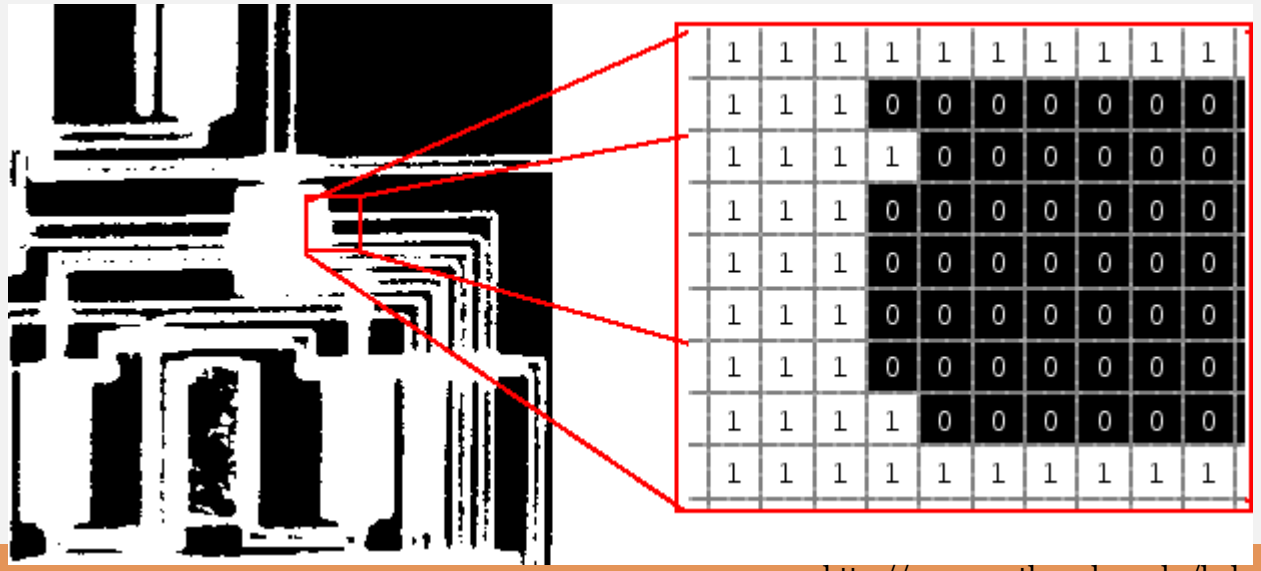
- 한글 문자 코드 표준
  - 조합형
  - 완성형
  - 완성형 확장
  - 유니코드
    - 한글 11,172자
    - 전세계 문자

| 연도 | 규격 번호 | 내용 |
|---|---|---|
| 1974년 | KSC 5601-1974 | 한글 자모 51자에 코드 부여 |
| 1977년 | KSC 5714-1977 | 한자 7,200자에 코드 부여 |
| 1982년 | KSC 5601-1982 | 2바이트 조합형 |
| | KSC 5619-1982 | 완성형한글 1,316자, 한자 1,692자 |
| 1987년 | KSC 5601-1987 | 완성형한글 2,350자, 한자 4,888자, ISO 2022 규격에 준함. |
| 1991년 | KSC 5601-1991 | 완성형 확장 글자한글 1,930자, 옛한글 1,673자, 한자 2,865자, KS C5601-1987을 보충하기 위하여 추가로 수집한 글자들 |
| 1992년 | KSC 5601-1992 | 2바이트 조합형을 완성형 한글과 함께 복수 표준화 |
| 1995년 | KSC 5700-1995 | 완성형 한글 11,172자, 조합형 자모 334자 및 23,274자의 한자 당시 유니코드 표준을 국가 표준화한 것임. |
| 1998년 | 최종 개정 | 신규격 번호 KS X 1001KS C 5601-1987 완성형의 신규격 |
| 2001년 | 최종 개정 | 신규격 번호 KS X 1001KS C 5657-1991의 신규격 |
| 2002년 | 최종 개정 | KS X 1005-1KS C 5700-1995(유니코드)의 신규격 |

http://news.korean.go.kr/online/see/hangulstory/hangulstory.jsp?idx=29#

# 멀티미디어 데이터의 표현

- **멀티미디어**
  - **이미지, 동영상, 음악, 소리 등**
  - 0,1 **이진수로 코딩하여 표현**
  - **여러가지 표준**
    - **이미지** : *.bmp, *.jpg
    - **동영상**: *.avi, *.mpg, *.mov
    - **음악** : *.mp3, *.wma




http://www.mathworks.co.kr/help/toolbox/images/f14-13543.html

# C 언어

- C언어의 소개
  - 1972년 AT&T Bell 연구소의 Dennis Ritchie와 Ken Thompson이 UNIX 운영체제의 개발에 사용할 목적으로 개발
  - C 언어의 상당부분은 그의 선조격인 CPL, BCPL, B Language 등의 영향을 받음
  - 객체지향 언어인 C++로 확장 (C#, Java에 영향)
  - 범용(general-purpose) 프로그래밍 언어, 시스템 프로그래밍에 적합
  - 효율성 있는 언어로 코드의 분량이 작고 빠르게 실행
  - 이식성(portability)이 좋은 언어
  - UNIX 운영체제와의 인터페이스를 제공하며 하위 수준의 하드웨어 제어가 용이함 (포인터 사용)
  - 1988-1990년에 ANSI C Standard와 ISO C Standard 제정
    - C 언어와 C 라이브러리의 표준화
    - 개발된 프로그램의 호환성 증대와 교육의 용이함 등의 잇점 제공

# C 프로그램 예제

```
#include <stdio.h>

int main()
{
  printf("Welcome to C Programming World!");

  return 0;
}
```

# 요점 정리

- 부호가 있는 정수를 표현하는 대표적인 방법으로 2의 보수 표현 방법이 있다.
- 사용하는 메모리의 크기에 따라 나타낼 수 있는 수의 범위가 결정된다.
- 표현가능한 범위를 벗어나면 잘못된 결과를 얻게 되므로, 표현 범위를 고려하여 데이터의 종류를  사용해야 한다.
- 소수점이 있는 수는 부동소수점수 표현 방법인 IEEE 754 표준을 따른다.
- 영문자는 ASCII 코드를 사용한다.
- 한글은 2바이트를 사용하여 표현되고, 표준으로 완성형확장과 유니코드 표준이 있다.
- 이미지, 동영상, 음악 등의 멀티미디어 데이터도 0, 1의 이진수로 표현되는 코드화 방법을 사용하여 표현된다.

# 요점 정리

- 정수형 데이터를 나타내는 자료형으로 char, short, int, long이 있다.
- 양수만 있는 정수 자료형에는 unsigned를 붙인 char, short, int, long이 있다.
- 영문자를 나타내는 문자형 데이터를 나타내는 자료형으로 char가 있다.
- 부동소수점 수 데이터를 나타내는 자료형으로 float, double이 있다.

# 다음 강좌 내용

- 변수와 자료형
  - 변수의 정의 방법 및 특성
  - 자료형
  - C 언어를 이용한 입출력