✳ 값 ⟶ 2진수

② 부동소수점 ⟶ 2진수 규칙 (IEEE 754)

* 현재 부동소수점을 2진수로 표현하는 유일한 방식

트리플트

값 ⟶ 2진수

2진수로 표현 가능하면 HDD와 RAM에 저장할 수 있음

12.375

먼저 2진수로 ⇒ 표현

1100 . 011

1100.011

정규화

같은 값

정해진 규칙에 따라서 정리함 왼쪽에 숫자 하나만 남겨두기

0.375

* 2

0.750 ⇒ 0 배서 기록

* 2

1.500 ⇒ 1 뺐으니 0.500을 인셈

* 2

1.000 ⇒ 1

0 밖에 안남았으니 끝!

2진수 정규화 ⇒ $1.100011 * 2^{3}$ (3: 지수부)

가수부

Excess-k ⇒ 3 + 127 = 130

지수부 + 127 = 130

그대로 | sign-magnitude

부호 절대값으로 저장

32bit 메모리 ⇒

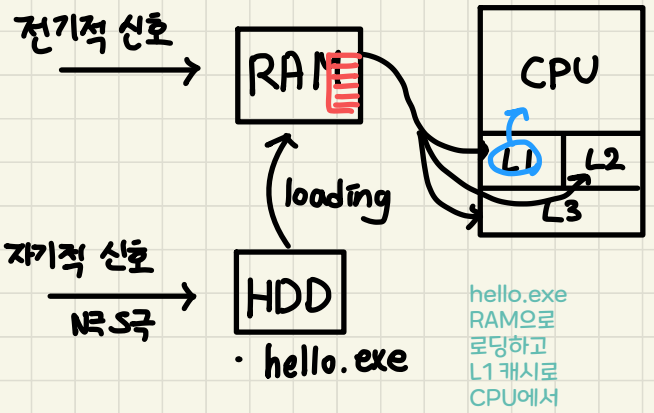
0	1	0	0	0	0	0	1	0	1	0	0	0	1	1	0	0	. . . .	0	. . . .

(±) 부호비트

지수부 (8bit)

가수부 (23bit)

0x41460000



EX) 정규화 예시 왼쪽에 숫자 하나만 남겨두기

314.592

↓

$3.14592 * 10^{2}$

0.00472

↓

$4.72 * 10^{-3}$

```
static float test = 12.375f;
static double test2 = 12.375;
```

987654321.1234567

입력

9.87654340E8

$\times 10^8$

987654340

아예 오류

부동소수점은 원래 정확한 값이 나오는 게 아님!

부동소수점 유효자리수

32byte	64 byte
거의 7자리까지	15자리까지 (16) 거의

* 부동소수점 값과 메모리 크기

32 bit 메모리

⇒ [1] [8] [23]

부호비트 ↑ 지수부 ↑ 가수부

지수부 7자리 → 둥치기

32bit : 7~8자리

64bit : 16자리

⇓

유효자릿수 범위의
부동 소수점을
2진수로 변환하면
최소의 오차로 값을 저장할 수 있다

JVM은 값을 꺼낼 때
최소의 오차값을 버리거나
적정한 위치에서 반올림 처리한다

-?~+? 소수점 이상 몇 개?

소수점 이하 몇 개?

IEEE 754 규칙에 따라
2진수로 변환하기 때문에
소수점 이상/이하 자릿수는
명시할 수 없다.

소수점을 제외한 숫자의 자릿수로 가능 범위를 표현
(맨 앞의 0도 제외한)

⇓

"유효 자릿수"라 부른다

* 값 ⟶ 2진수

③ 문자 ⟶ 2진수규칙

1) ASCII (7bit) → 국제표준 X

"국제표준" + 유럽문자

2) ISO-8859-x (8bit) ← ASCII 포함

A →	0100 0001	(0x41)
B →	0100 0010	(0x42)
C →	0100 0011	(0x43)

⋮

"ISO-latin-1"

S
15

A →	100_4	0001_1
B →	100_4	0010_2
C →	100_4	0011_3
a →	110_6	0001_1
b →	110_6	0010_2
c →	110_6	0011_3
0 →	110_6	0011_0
1 →	011_3	0001_1
2 →	011_3	0010_2
? →	011_3	1111_F
공백 →	010_2	0000_0

국제표준

3) EUC-KR (16bit) ← 최대 65536자 → 4) 조합형 (16bit)

국제표준 X

← 모든 한글 가능

↳ 한글 2350자 + α

↳ 영어는 ISO-8859-1 사용

완성된 문자에 대해 2진수를 정의 ⇓ 완성형

- 가 → 0xB0A1 (1011 0000 1010 0001)
- 각 → 0xB0A2
- 돌 → 0x
- 똠 → ?
- 똡 → ?
- 똥 → 0x8bcB

⇒ Windows 3.1까지

1000 1000 0110 0001

8 8 6 1

조합형 (16bit)

초성 : ㄱ ㄲ ㄴ ㄷ ㄸ ㄹ ···

(2 3 4 5 6 7 ⇒ 5bit 사용)

중성 : ㅏ ㅐ ㅑ ㅒ ㅓ ··

(3 4 5 6 7 ⇒ 5 bit 사용)

종성 : ㄱ ㄲ ㄳ ㄴ ㄵ ㄶ ㄷ ····

(2 3 4 5 6 7 8 ⇒ 5bit 사용)

한글에 붙임

닫 → 1 | ㄷ 00101 | ㅏ 00011 | ㄷ 01000

9 4 6 8

0x

⇓

0x9468

HWP에디터에서 사용

* 똠똥 입력도 못하는데 MS Word를 공공기관에서 어떻게 써? 할 수 없다.. MS949 만들음

← 국제표준 X

5) MS949 (CP949) (16bit) ⟶

= EUCKR (2350자+) + α

돌 → B6CA (EUC-KR)

똠 → 8C63 (MS949)

뚑 → 8C64 (MS949)

똥 → 86CB (EUC-KR)

↳ 기존 EUC-KR은 그대로 사용

⇓

(Windows 95에서 추가됨) 기존에 작성한 문서는 그대로 사용 가능

↳ 문자 규칙 추가

⇓

출판 업계에서 유통되고 있는 문자를 대부분 지원 (11172자)

* Windows 10은 기본적으로 문서가 MS949로 저장됐을 것이라 간주함 ⇒ encoding 필요함

6) Unicode → UTF-16 (16bit)

A → 0x0041
B → 0x0042
C → 0x0043
⋮ 2byte

} 영어 대소문자 숫자

→ 2byte로?!

가 → 0xAC00
각 → 0xA01
⋮ 2byte

} 한글은 현대한글음절 11172자를 새로 정의

⇓

기존 규칙과 호환되지 않는다.

기존 파일과 호환X

⇓

Java에서 문자를 다룰 때 이 규칙을 사용한다

1 byte - 8bit

7) UTF-8 (8 ~ 32bit) ← Unicode 변형 포맷 [국제표준]
1~4byte
↳ ASCII

Unicode (Universal Coded Character Set)

Transformation

Format

↳ 기존 ASCII 규칙을 그대로 사용하기 위해 만든 규칙
영어로↓
이전에 작성한 문서를 그대로 사용할 수 있다.

* 한글은 한자에 기본 3byte (24bit)
영어는 1byte (8bit)

* 예) 가

MS949 (EUC-KR+α) : B0A1 (2byte)
조합형 : 8861 (2byte)
Unicode (UTF-16) : AC00 (2byte)
1010 1100 0000 0000

UTF-8:
1110 1010 1011 0000 1000 0000
E A B 0 8 0

0000 ~ 007F(ASCII) → 그대로 1Byte로 표현
0080 ~ 07FF → 110 xxxxxx 10xx xxxx (2 byte). 유럽문자
0800 ~ FFFF → 1110 xxxx 10xx xxxx 10xx xxxx (3 byte). 한글 등
10000 ~ —— → 4byte

1 byte = 8 bit

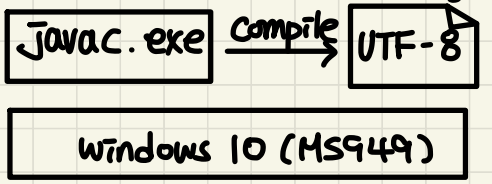
* 맥(unix) UTF-8 국제표준
Windows 사용자는 필수로
source를 UTF-8로 저장해야함

* 문자 ⟶ 2진수 규칙

↳ Character set (문자 집합) ← * 문자를 2진수로 바꾸는 규칙

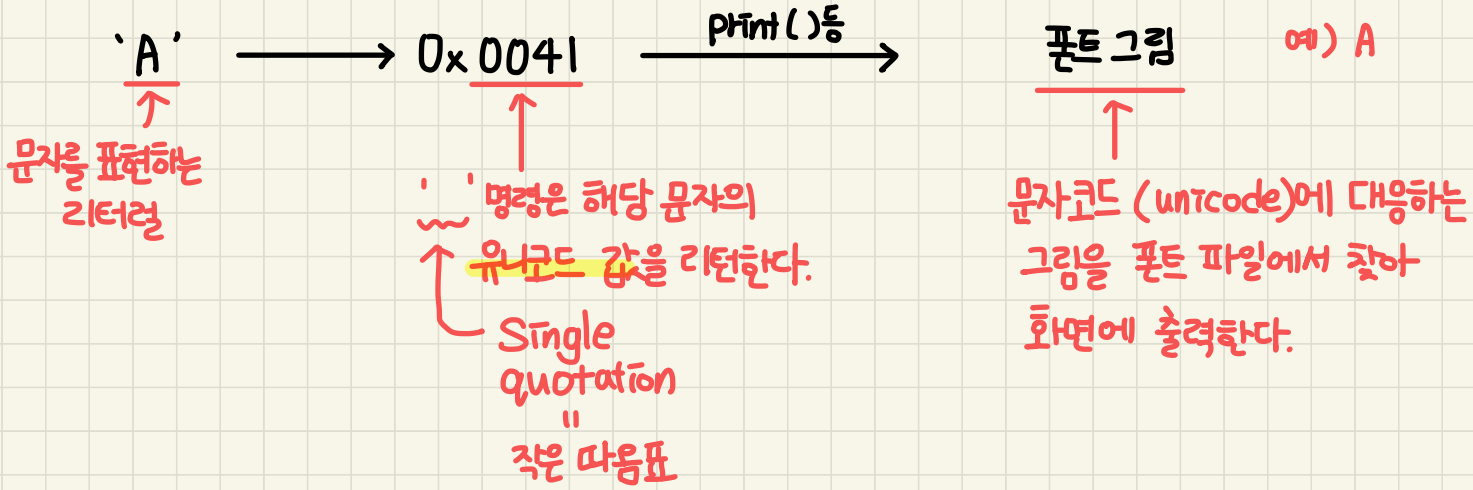
- ASCII (7bit)
- ISO-8859-1 (8 bit) (ISO-latin-1)
- EUC-KR (KS C-5601) (2 byte)
- 조합형 (2byte)
- MS949 (CP949) (2 byte)
- Unicode (UTF-16) (2 byte)
- UTF-8 (1 ~ 4 byte)

$ javac -encoding UTF-8 Hello.java



* 폰트 파일

↳ 문자 코드에 대응하는 문자그림이 들어 있는 파일

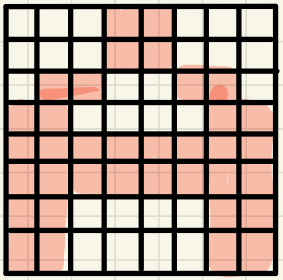


* 폰트 파일의 유형

① Raster 폰트 (bitmap)

↳ 점으로 폰트를 그린 것

Image의 경우:
비트맵 이미지
사진 이미지
⋮
래스터 방식
⇓
사진크기를 늘리면
⇓
계단현상



↳ 글자 그림을 그대로 출력
↳ 출력 속도가 빠르다

↳ 글자 크기를 키우면
↳ 계단 현상 발생

② Vector 폰트 (Truetype)

↳ 폰트를 그리는 명령

예)
(1,4)(1,5) 직선
(2,4)(2,5) 직선
(3,2)(4,1) 대각선
⋮

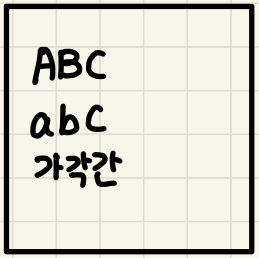
↳ 명령어에 따라 그림을 그린다
↳ 출력 속도가 느리다

↳ 글자 크기를 늘려도
계단 현상이 발생하지 않는다.

예) TTF (True Type Font)

Image의 경우:
(CAD 이미지
클립아트
⇓
그림의 크기를
늘려도
계단현상 X

✳ 줄바꿈 코드값



편집기

코드값

A	B	C	줄바꿈		a	b	c		가	각	간	
41	42	43	0D	0A	61	62	63	0D0A	EA8080	EA8081	EA8084	0D0A

- 0D: Carrage Return (CR)
- 0A: Line Feed (LF)
  - ↳ Unix 에서는 0A 만 있음
- (CRLF)
- git default : CRLF

* eclipse는
\r을 줄바꿈으로 인식
↳ 확인하려면
cmd 창으로 확인
\f도 안먹힘 (무서!)

✳ " " double quotation
안에 "를 삽입하고
싶을 때는 \ 입력

\\Users\\user
↳ 이스케이프 문자가 아냐!

* java-basic

  .settings / .classpath / .project } 삭제하고 다시 gradle eclipse