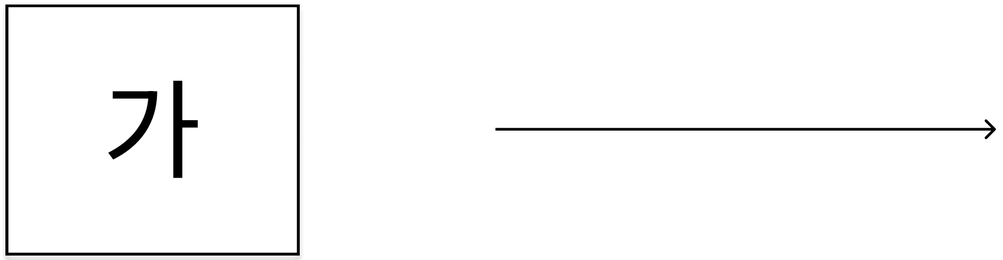
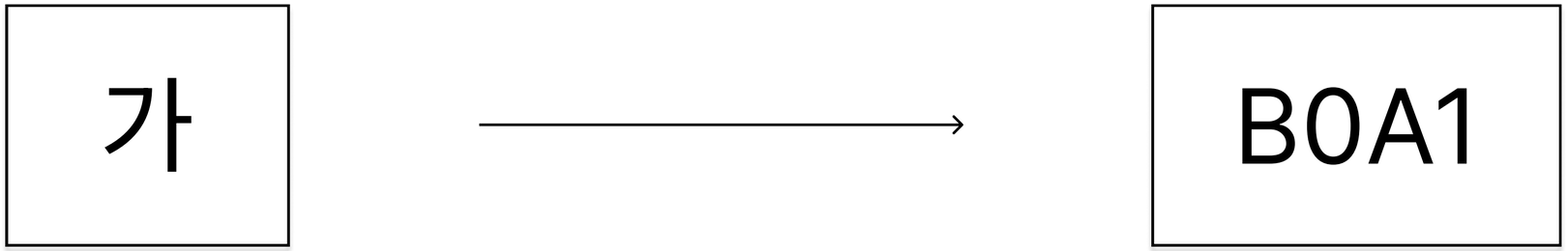
# 컴퓨터가 문자를 이해하는 방법

# 컴퓨터는 한글을 어떻게 인지할까?

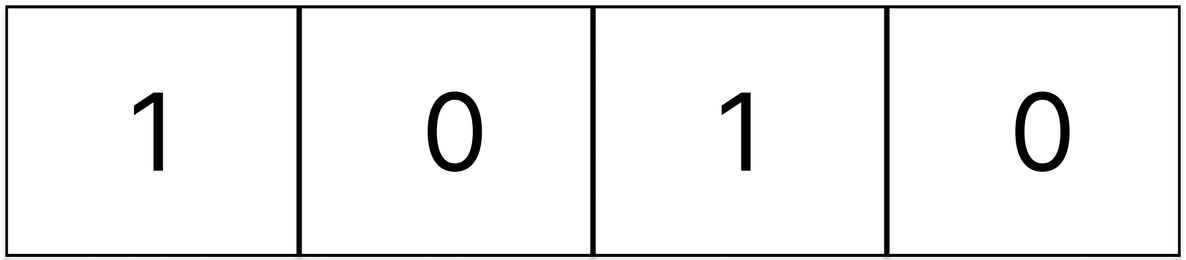


# 컴퓨터는 한글을 어떻게 인지할까?



# 비트 (bit)

4bit

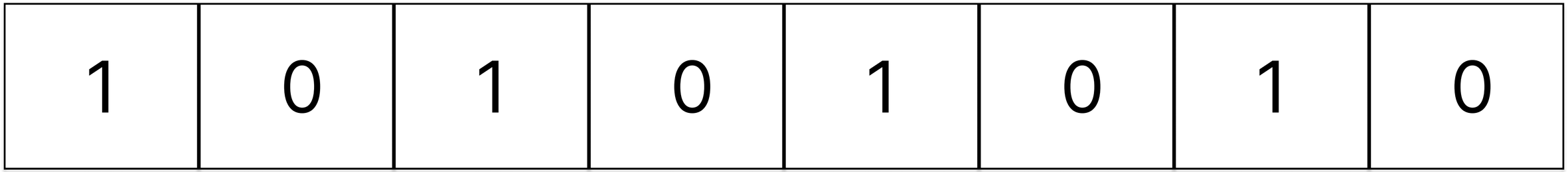



1bit 1bit 1bit 1bit

1이나 0을 저장할 수 있는 단위

# 바이트 (byte)

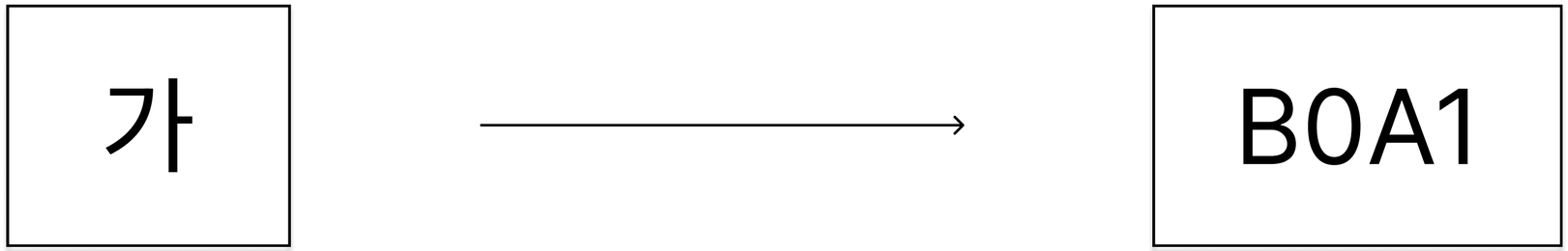
8bit = 1byte




1bit 1bit 1bit 1bit 1bit 1bit 1bit 1bit

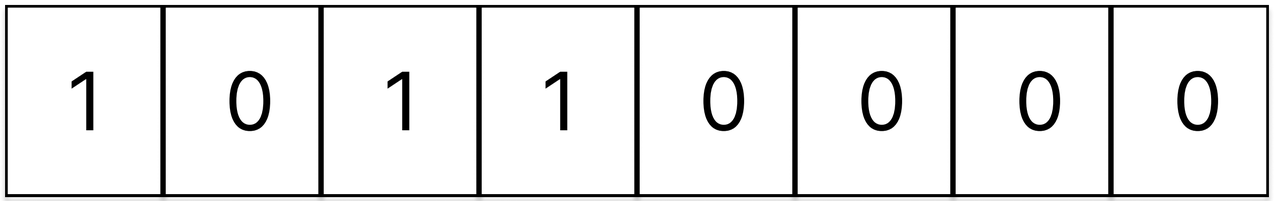
8개의 bit를 묶어 256개 까지 표현하는 단위

# 컴퓨터는 한글을 Byte코드로 인지한다.

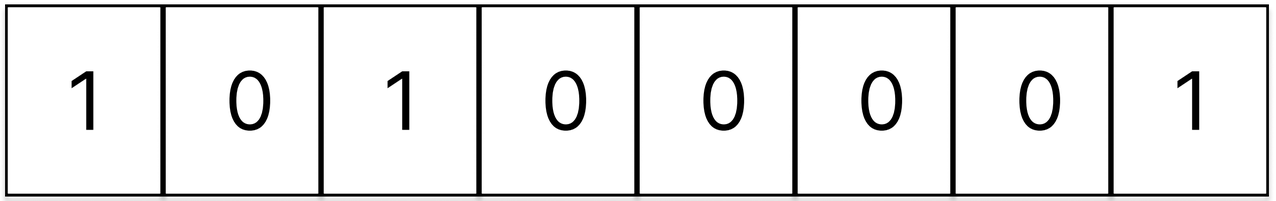



2btye

# 2진수와 16진수




= B0




= A1

“가”

# 인코딩

# 인코딩의 방법

EUC-KR MS949 UTF-8

# EUC-KR

## 한글을 2byte로 표현

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| b0a0 | | 가 | 각 | 간 | 갇 | 갈 | 갉 | 갊 | 감 | 갑 | 값 | 갓 | 갔 | 강 | 갖 | 갗 |
| b0b0 | 같 | 갚 | 갛 | 개 | 객 | 갠 | 갤 | 갬 | 갭 | 갯 | 갰 | 갱 | 갸 | 갹 | 갼 | 걀 |
| b0c0 | 걋 | 걍 | 걔 | 걘 | 걜 | 거 | 걱 | 건 | 걷 | 걸 | 걺 | 검 | 겁 | 것 | 겄 | 겅 |
| b0d0 | 겆 | 겉 | 겊 | 겋 | 게 | 겐 | 겔 | 겜 | 겝 | 겟 | 겠 | 겡 | 겨 | 격 | 겪 | 견 |
| b0e0 | 겯 | 결 | 겸 | 겹 | 겻 | 겼 | 경 | 곁 | 계 | 곈 | 곌 | 곕 | 곗 | 고 | 곡 | 곤 |
| b0f0 | 곧 | 골 | 곪 | 곬 | 곯 | 곰 | 곱 | 곳 | 공 | 곶 | 과 | 곽 | 관 | 괄 | 괆 | |
| b1a0 | | 괌 | 괍 | 괏 | 광 | 괘 | 괜 | 괠 | 괩 | 괬 | 괭 | 괴 | 괵 | 괸 | 괼 | 굄 |
| b1b0 | 굅 | 굇 | 굉 | 교 | 굔 | 굘 | 굡 | 굣 | 구 | 국 | 군 | 굳 | 굴 | 굵 | 굶 | 굻 |
| b1c0 | 굼 | 굽 | 굿 | 궁 | 궂 | 궈 | 궉 | 권 | 궐 | 궜 | 궝 | 궤 | 궷 | 귀 | 귁 | 귄 |
| b1d0 | 궐 | 귐 | 귑 | 귓 | 규 | 균 | 귤 | 그 | 극 | 근 | 귿 | 글 | 긁 | 금 | 급 | 긋 |
| b1e0 | 긍 | 긔 | 기 | 긱 | 긴 | 긷 | 길 | 긺 | 김 | 깁 | 깃 | 깅 | 깆 | 깊 | 까 | 깍 |

EUC-KR 코드표 일부

## 2350개

# MS949

EUC-KR의 확장판

EUC-KR ⇒ MS949

“횡붹”

# MS949

EUC-KR의 확장판

EUC-KR ⇒ MS949

“??”

# MS949

MS949 ⇒ EUC-KR

“횡붹”

# MS949

MS949 ⇒ EUC-KR

“??”

MS949

# 확장이라면서 왜?

# MS949

횡, 이나 붾은 EUC-KR은 표현할 수 없다

# MS949

1. 힣 or 뷁 ⇒ EUC-KR ⇒ “???” ⇒ MS949

이미 byte배열이 깨져있다

2. 힣 or 뷁 ⇒ MS949 ⇒ 힣, 뷁 ⇒ EUC-KR

결국 EUC-KR이 읽지 못함

# 인코딩 방식의 중요성

IF...

레거시 DB ⇒ US7ASCII, EUC-KR

백엔드 ⇒ UTF-8, MS949

웹 ⇒ UTF-8

제대로 읽어올 수 있을까?

# UTF-8

“유니코드 문자열(Charset)”

# UTF-8


유니코드 표

코드 포인트에 U+ (lg 아님 ㅋ)붙여서 사용

# UTF-8

| UTF-8 | UTF-16 | UTF-32 |
| --- | --- | --- |

저장하는 **byte** 크기의 차이

# UTF-8

| 2~4byte | 2, 4byte | 4byte |
| --- | --- | --- |
| UTF-8 | UTF-16 | UTF-32 |

저장하는 **byte** 크기의 차이

# UTF-8

| 코드포인트 범위 | 인코딩 방법 | 크기 |
|---|---|---|
| U+0000 ~ U+007F | 그대로 인코딩 | 1 byte |
| U+0080 ~ U+07FF | 110xxxxx 10xxxxxx | 2 byte |
| U+0800 ~ U+FFFF | 1110xxxx 10xxxxxx 10xxxxxx | 3 byte |
| U+10000 ~ U+1FFFFF | 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx | 4 byte |

[ 표 ] 코드포인트 범위에 따른 UTF-8 인코딩 방법

가 ⇒ U+AC00　1010110000000000

11101010101100001000000(3byte)

끝