2018학년도 2학기 언어와 컴퓨터

제9강 문자 인코딩

박수지

서울대학교 인문대학 언어학과

2018년 10월 10일 수요일

오늘의 목표

1. 문자 인코딩이 무엇인지 설명할 수 있다.
2. 한글 인코딩에서 겪었던 문제가 무엇인지 말할 수 있다.
3. 유니코드의 특징을 말할 수 있다.

부호화-복호화 모형 (Encoding-decoding model)

주요 개념

부호화 정보를 다른 형태로 변환하는 처리

복호화 부호화된 형태를 원래 정보로 복원하는 처리

부호 정보 변환 규칙 체계

코덱 부호화·복호화를 수행하는 기계나 알고리듬

목적

표준화, 보안, 압축 등

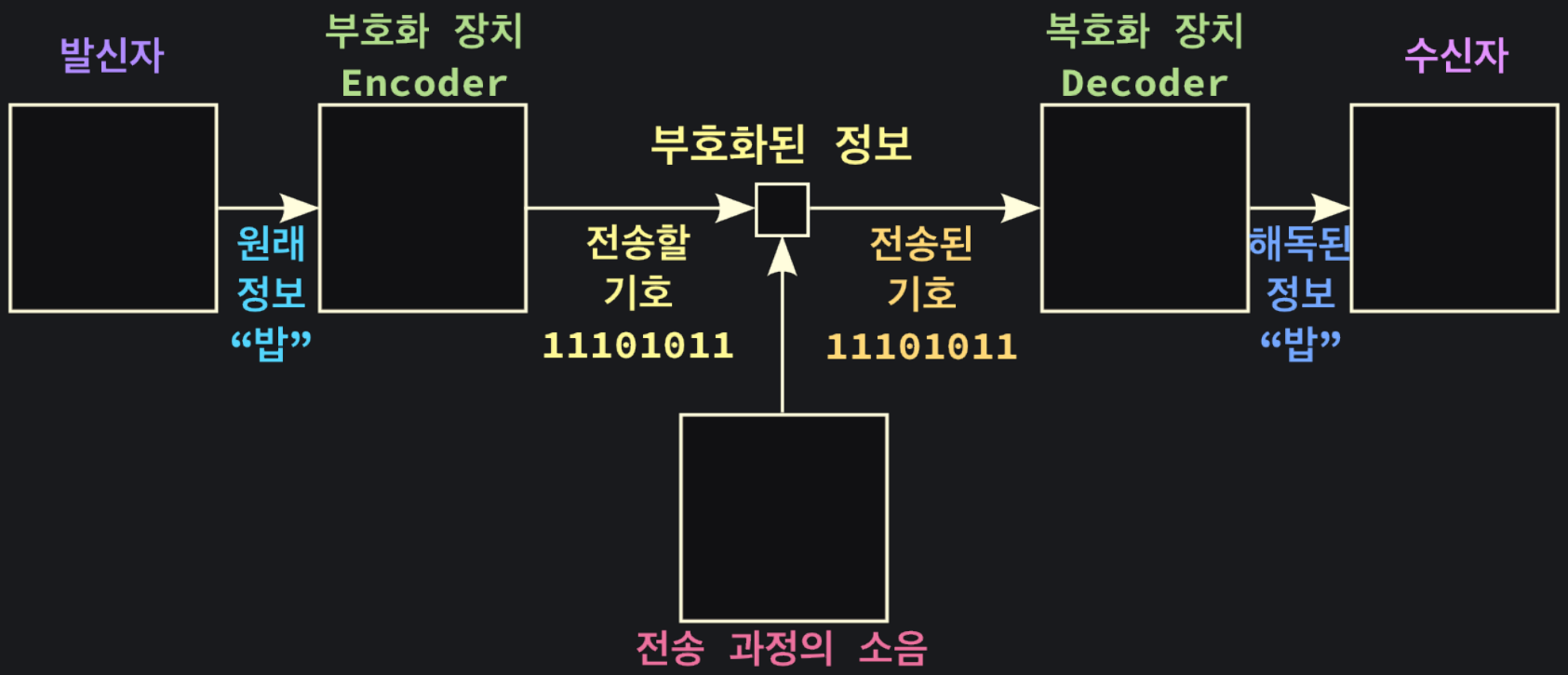
부호화 예시

(주로 인간용) $\rightarrow$ (주로 기계용)

- 알파벳 $\rightarrow$ 모스 부호
- 집 $\rightarrow$ 주소
- 음악 $\rightarrow$ 악보
- 이미지 $\rightarrow$ 픽셀
- 평문 $\rightarrow$ 암호문

부호화-복호화 모형 (Encoding-decoding model)

통신 이론



Shannon. (1948). “The Mathematical Theory of Communication”. The Bell System Technical Journal 27, 379–423.

https://commons.wikimedia.org/wiki/File:Shannon_communication_system.svg
http://math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf

오늘의 부호

문자 코딩: 문자 $\rightarrow$ N 바이트 이진수

예시 '가' $\rightarrow$ `11101010 10110000 10000000`

1바이트

= 8비트
= 2진수 여덟 자리
= 2진수 네 자리 두 개
= 16진수 두 개

표현 예시

10진수 111
2진수 1101111
1바이트 0110 1111
16진수 6 F

16진수

10 A
11 B
12 C
13 D
14 E
15 F

표현 가능한 가짓수

1바이트 $2^8 = 256$

2바이트 $2^{16} = 65536$

3바이트 $2^{24} = 16777216$

문자 개수

숫자 10

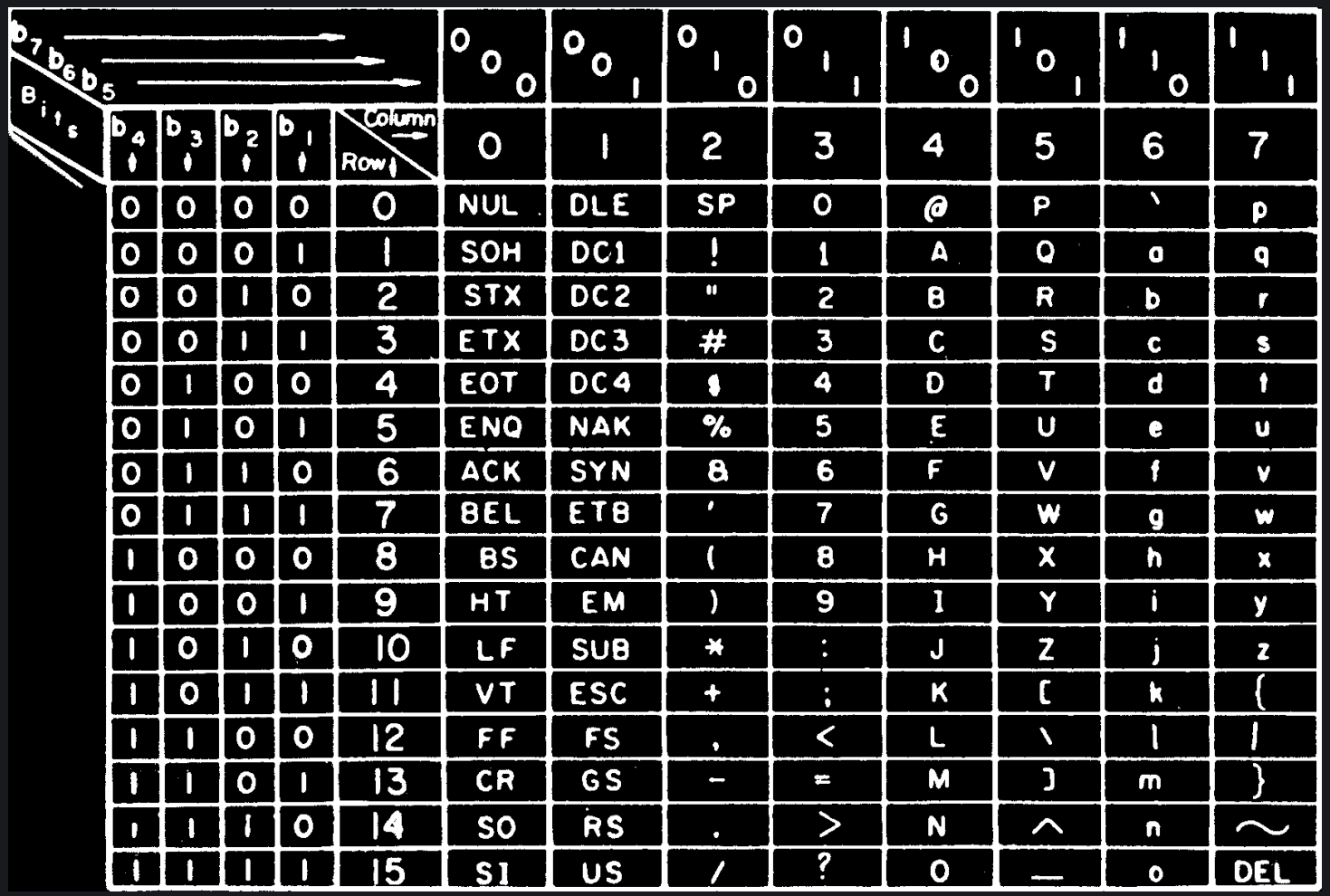
로마자 $26 \times 2 = 52$

한글 $19 \times 21 \times 28 = 11172$

한자 106230 (異體字字典)

ASCII(American Standard Code for Information Interchange)

7비트로 영문자, 숫자, 특수 문자, 공백 문자를 표현하는 인코딩 방식

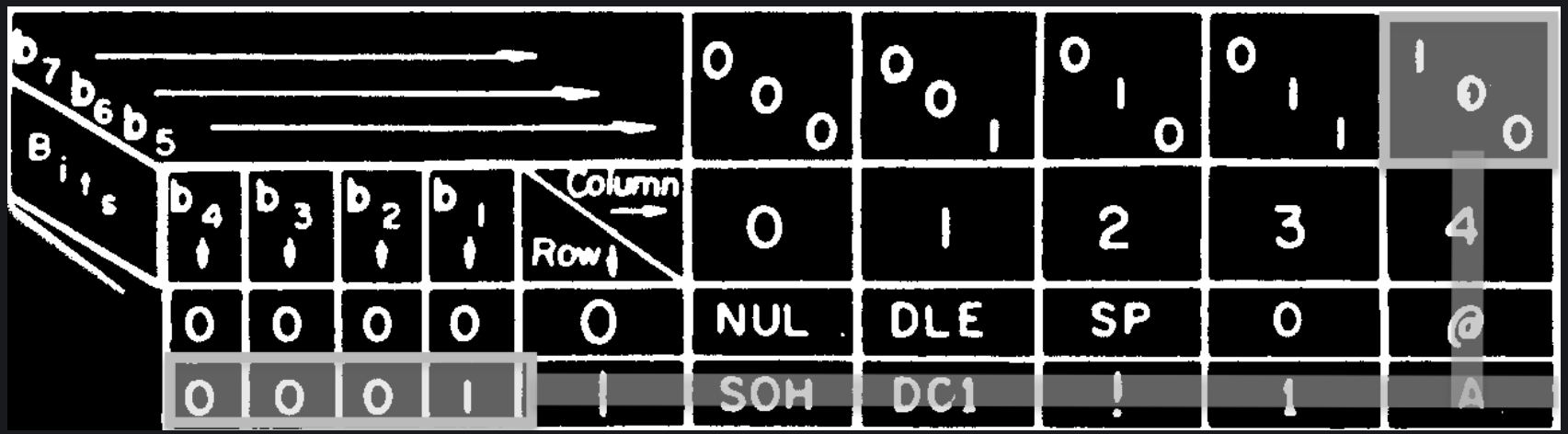


b_7 b_6 b_5 →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Bits b_4 b_3 b_2 b_1 ↓ / Column → / Row ↓	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
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	FS	,	<	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

https://commons.wikimedia.org/wiki/File:USASCII_code_chart.png

아스키 코드에서 A에 대응하는 7비트 값 찾기

`100 0001` (열 `100`; 행 `0001`)



1바이트 완성하기

A `100 0001` → `0100 0001`

B `100 0010` → `0100 0010`

C `100 0011` → `1100 0011`

패리티 비트 (parity bit)

- 1이 짝수 개 → 0
- 1이 홀수 개 → 1

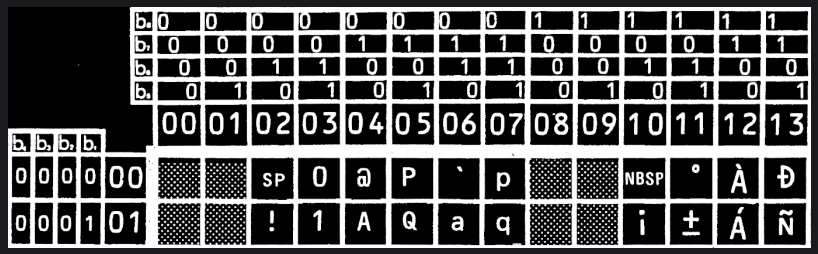
확장 아스키 코드

- 8비트($2^8 = 256$가지)를 모두 사용하여 수학 기호($\times$, $\geq$, π) 및 확장 로마자(é, ç) 등을 1바이트로 표현하는 인코딩 방식

  ISO 8859-1 서유럽
  ISO 8859-2 동유럽
  ISO 8859-3 남유럽
  ISO 8859-4 북유럽
  ISO 8859-5 키릴 문자
  … …

ISO 8859-1 인코딩 예시

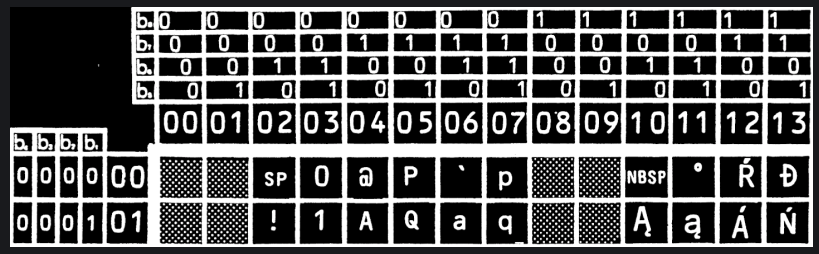
Ñ → 1101 0001



ISO 8859-1

ISO 8859-2 인코딩 예시

Ń → 1101 0001



ISO 8859-2

http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-094.pdf

파이썬에서 인코딩하기

```
1 from unicodedata import lookup
2 lookup('LATIN␣CAPITAL␣LETTER␣N␣WITH␣TILDE').
    encode('ISO␣8859-1')
3 lookup('LATIN␣CAPITAL␣LETTER␣N␣WITH␣ACUTE').
    encode('ISO␣8859-2')
```

파이썬에서 디코딩하기

```
1 b'\xd1'.decode('ISO␣8859-1')
2 b'\xd1'.decode('ISO␣8859-2')
```

한글 인코딩의 문제

- 한글 글자 수는 $11,172$로 $2^{13} = 8,192$보다 크다.

$\Rightarrow$ 1바이트로 표현할 수 없다.

한글 인코딩 방법

N바이트 조합형, 3바이트, 7비트 완성형, 2바이트 조합형, 2바이트 완성형, 확장 완성형, 유니코드(UTF-8, UTF-16, UTF-32), ...

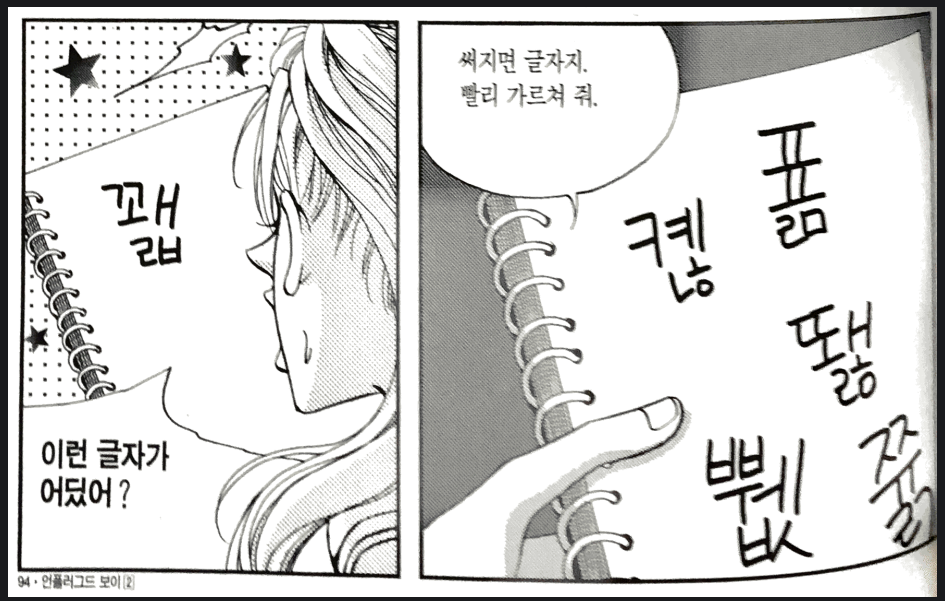
다양한 한글 코드

완성형 (KSC 5601)

한계

한글 2,350자만 표현 가능

- **똠**방각하
- **폞**시콜라
- 설**믜**
- **꽯, 켾, 푦, …**



천계영. 언플러그드 보이 2. 서울문화사. 1997.

다양한 한글 코드

완성형 (KSC 5601)

‘김설의’라는 이름이 입력되지 않는 이유는 은행이나 통신사, 대학 등 민간에서 사용하는 대형 전산시스템의 한국산업표준(KS)이 ‘한글조합형코드’가 아닌 ‘한글완성형코드’(EUC-KR)인 탓이다. 완성형코드는 국제표준과 충돌이 적다는 장점이 있지만, 미리 조합되어 있는 글자 외의 문자는 인식할 수 없다는 단점이 있다. ‘믜’, ‘뀛’ 같이 빈도수가 낮은 문자들은 코드에 등록하지 않아 문자로 보지 못하는 셈이다. 이 완성형코드는 한글 초·중·종성으로 조합 가능한 한글 문자 1만1172자 중 2350자만 표현할 수 있다.

http://www.hani.co.kr/arti/society/society_general/864914.html

인코딩 예시

```
>>> '믓'.encode('euc-kr')
b'\xb9\xcb'
>>> '미'.encode('euc-kr')
b'\xb9\xcc'
```

‘믕’, ‘믖’, …, ‘믜’ 등이 없다!

다양한 한글 코드

확장 완성형 (Microsoft Unified Hangul)

의의

2,350자 외에도 코드가 배당되었다.

문제

코드 순서가 가나다순이 아니다.

- 가나다순 정렬이 불가능하다.
- 자소 분해가 불가능하다.

인코딩 예시

```
>>> '뭇'.encode('cp949')
b'\xb9\xcb'
>>> '믜'.encode('cp949')
b'\x92\xde'
>>> '미'.encode('cp949')
b'\xb9\xcc'
```

'믜'가 있지만 '뭇'과 '미' 사이가 아니다!

다양한 한글 코드

북한의 표준 문자 코드 (KPS 9566)

특징

- 북한의 자모순으로 배열
- 한글 특수문자 존재

행 \ 렬	72	73	74	75	76	77	78	79	80	81
1	⌒	⌣	⌃	⌄	⩔	⩓	⌐	⌙	⌐	⌙
2	▽	▼	▷	◁	▶	◀	○	●	❖	◢
3	h	i	j	k	l	m	n	o	p	q
4	김	일	성	김	정	일				
5	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
6	Ⅷ	Ⅸ	Ⅹ							i
7	⁸	⁹	½	⅓	⅔	¼	¾		₀	₁
8	Pa	kPa	MPa	GPa	ℓ	μℓ	mℓ	dℓ	kℓ	gal

박진호. 국어 정보화의 방향: 문자 코드를 중심으로. 새국어생활 25-2. 2015.

한글이 깨지는 이유

부호화 방법과 복호화 방법이 일치하는 경우

```
>>> '고기'.encode('utf-8').decode('utf-8')
'고기'
>>> '고기'.encode('euc-kr').decode('euc-kr')
'고기'
```

부호화 방법과 복호화 방법이 일치하지 않는 경우

```
>>> '고기'.encode('utf-8').decode('euc-kr', 'ignore')
'怨湲'
>>> '고기'.encode('euc-kr').decode('utf-8', 'replace')
'□□□□'
```

유니코드

모든 문자를 부호화·표현·처리하는 산업 표준

Unicode Consortium(`https://unicode.org`)

특징

- 한글 11,172자가 가나다순으로 배열
- 옛한글 자모 포함
- 아스키 코드에 존재하는 문자는 아스키 코드와 같은 포인트에 대응

인코딩 방식

UTF-8, UTF-16, UTF-32 등

유니코드

차트 구성

A960 **Hangul Jamo Extended-A** **A97F**

	A96	A97
0	ㄸㅁ A960	ㅍㅊ A970
1	ㄸㅐ A961	ㅍㅋ A971
2	ㄸㅅ A962	ㅍㅌ A972
3	ㄸㅈ A963	ㅍㅎ A973
4	ㄹㄱ A964	ㅎㄱ A974

Old initial consonants

A960	ㄸㅁ	HANGUL CHOSEONG TIKEUT-MIEUM
A961	ㄸㅐ	HANGUL CHOSEONG TIKEUT-PIEUP
A962	ㄸㅅ	HANGUL CHOSEONG TIKEUT-SIOS
A963	ㄸㅈ	HANGUL CHOSEONG TIKEUT-CIEUC
A964	ㄹㄱ	HANGUL CHOSEONG RIEUL-KIYEOK
A965	ㄹㄲ	HANGUL CHOSEONG RIEUL-SSANGKIYEOK
A966	ㄹㄷ	HANGUL CHOSEONG RIEUL-TIKEUT
A967	ㄹㄸ	HANGUL CHOSEONG RIEUL-SSANGTIKEUT
A968	ㄹㅁ	HANGUL CHOSEONG RIEUL-MIEUM
A969	ㄹㅐ	HANGUL CHOSEONG RIEUL-PIEUP
A96A	ㄹㅃ	HANGUL CHOSEONG RIEUL-SSANGPIEUP
A96B	ㄹㅒ	HANGUL CHOSEONG RIEUL-KAPYEOUNPIEUP
A96C	ㄹㅅ	HANGUL CHOSEONG RIEUL-SIOS
A96D	ㄹㅈ	HANGUL CHOSEONG RIEUL-CIEUC
A96E	ㄹㅋ	HANGUL CHOSEONG RIEUL-KHIEUKH
A96F	ㅁㄱ	HANGUL CHOSEONG MIEUM-KIYEOK
A970	ㅁㄷ	HANGUL CHOSEONG MIEUM-TIKEUT
A971	ㅁㅅ	HANGUL CHOSEONG MIEUM-SIOS
A972	ㅂㅌ	HANGUL CHOSEONG PIEUP-SIOS-THIEUTH

https://unicode.org/charts/PDF/UA960.pdf

UTF-8로 한글 인코딩하기

형식: 3바이트 `1110XXXX 10XXXXXX 10XXXXXX`

- 빈칸 16자리 → 4자리 2진수 (=1자리 16진수) 네 개로 표현 가능
- 범위: `AC00–D7A3`

	AC0	AC1	AC2	AC3	AC4	AC5	AC6	AC7	AC8	AC9	ACA	ACB	ACC	ACD	ACE	ACF
0	가 AC00	감 AC10	갠 AC20	갰 AC30	걀 AC40	걬 AC50	걠 AC60	거 AC70	검 AC80	겐 AC90	겠 ACA0	결 ACB0	겼 ACC0	곀 ACD0	고 ACE0	곰 ACF0
1	각 AC01	갑 AC11	갢 AC21	갱 AC31	걁 AC41	걑 AC51	걡 AC61	걱 AC71	겁 AC81	겑 AC91	겡 ACA1	겱 ACB1	곁 ACC1	곡 ACD1	곡 ACE1	곱 ACF1
2	갂 AC02	값 AC12	갣 AC22	갲 AC32	걂 AC42	걒 AC52	걢 AC62	걲 AC72	겂 AC82	겒 AC92	겢 ACA2	겲 ACB2	곂 ACC2	곢 ACD2	곢 ACE2	곲 ACF2
3	갃 AC03	갓 AC13	갤 AC23	갳 AC33	걃 AC43	걓 AC53	걣 AC63	걳 AC73	것 AC83	겓 AC93	겣 ACA3	겳 ACB3	곃 ACC3	곣 ACD3	곣 ACE3	곳 ACF3
4	간 AC04	갔 AC14	갥 AC24	갴 AC34	걄 AC44	걔 AC54	걤 AC64	건 AC74	걷 AC84	겔 AC94	겤 ACA4	겴 ACB4	계 ACC4	곔 ACD4	곤 ACE4	곴 ACF4

예시: UTF-8로 '가' 인코딩하기

1. '가'에 대응하는 16진수 AC00
   - `AC00 = 1010 1100 0000 0000`
2. ...에 대응하는 바이트
   - `11101010 10110000 10000000`

https://unicode.org/charts/PDF/UAC00.pdf

파이썬에서 활용하기

문자 코드 포인트

```
>>> ord('가')
44032
>>> chr(ord('가'))
'가'
>>> chr(44032)
'가'
>>> hex(44032)
'0xac00'
```

문자 이름

```
>>> from unicodedata import name, lookup
>>> name('가')
'HANGUL SYLLABLE GA'
>>> lookup(name('가'))
'가'
>>> lookup('HANGUL SYLLABLE GA')
'가'
```

파이썬에서 활용하기

현대 한글 자모

초성(19개) ㄱ ㄲ ㄴ ㄷ ㄸ ㄹ ㅁ ㅂ ㅃ ㅅ ㅆ ㅇ ㅈ ㅉ ㅊ ㅋ ㅌ ㅍ ㅎ

중성(21개) ㅏ ㅐ ㅑ ㅒ ㅓ ㅔ ㅕ ㅖ ㅗ ㅘ ㅙ ㅚ ㅛ ㅜ ㅝ ㅞ ㅟ ㅠ ㅡ ㅢ ㅣ

종성(28개) ∅
ㄱ ㄲ ㄳ ㄴ ㄵ ㄶ ㄷ ㄹ ㄺ ㄻ ㄼ ㄽ ㄾ ㄿ ㅀ ㅁ ㅂ ㅄ ㅅ ㅆ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ ㅎ

다음 중성과의 거리

```
>>> ord('개') - ord('가')
28
>>> ord('월') - ord('울')
28
```

다음 초성과의 거리

```
>>> ord('까') - ord('가')
588
>>> ord('줄') - ord('울')
588
```

요약

문자 인코딩

문자를 N 바이트 이진수로 변환하는 규칙 체계

바람직한 한글 인코딩의 요건

- 현대 한글 11,172자를 모두 포함
- 자모순으로 배열

유니코드

세계의 모든 문자를 컴퓨터에서 통합된 체계로 표현하기 위한 표준

코드 포인트 `ord()` $\leftrightarrow$ `chr()`

문자 이름 `unicodedata.name()` $\leftrightarrow$ `unicodedata.lookup()`

다음 시간에 배울 것

- ‘한’을 인자로 받아 ‘ㅎㅏㄴ’을 반환하는 함수 만들기
- 옛한글 ‘ᄉᆞᆯ’ 입력하기

더 읽을 것

박진호. 국어 정보화의 방향: 문자 코드를 중심으로. 새국어생활 25-2. 2015.