Base58 인코딩 및 디코딩

■ Base58

- 비트코인의 익명 개발자이기도 한 **사토시 나카모토는 Base58을 개발했음**
- Base64와 유사한 수준의 데이터 압축률을 달성하는 동시에 0(0), O(대문자 o), I(대문자 i), l(소문자 L)과 유사한 문자를 제거하여 사람이 읽기 쉽게 만드는 것이 목표였음
    * 가독성을 위해 영숫자 +(더하기)와 /(슬래시)도 제거했음
    * Base64와 매우 유사하지만, 문자 수가 6개 적어 효율이 떨어짐
    * 또한, base가 2의 거듭제곱이 아니기 때문에 파싱이 약간 더 어려움

- 총 **58가지의 고유한 문자를** 사용하여 데이터를 표현
- 바이트(bytes) 데이터를 처리하므로, 인코딩 전에 문자열을 바이트로 변환해야 함(예: text.encode('utf-8')).

- **Base58의 문자 집합**
    * 혼동하기 쉬운 문자들을 의도적으로 제외하여 사용자 편의성과 오류 방지를 높였dma
    * **제외된 문자: 숫자 '0', 대문자 'O', 대문자 'I', 소문자 'l' (숫자 1로 오인될 수 있음)**

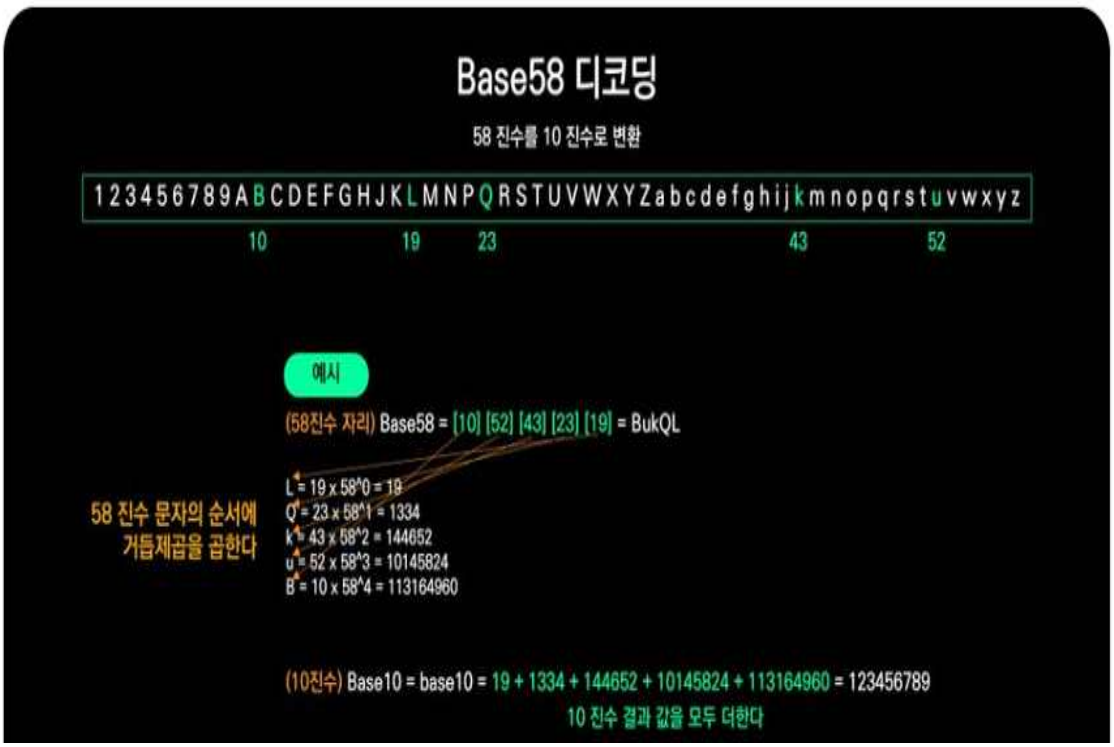
Value	Character	Value	Character	Value	Character	Value	Character
0	1	1	2	2	3	3	4
4	5	5	6	6	7	7	8
8	9	9	A	10	B	11	C
12	D	13	E	14	F	15	G
16	H	17	J	18	K	19	L
20	M	21	N	22	P	23	Q
24	R	25	S	26	T	27	U
28	V	29	W	30	X	31	Y
32	Z	33	a	34	b	35	c
36	d	37	e	38	f	39	g
40	h	41	i	42	j	43	k
44	m	45	n	46	o	47	p
48	q	49	r	50	s	51	t
52	u	53	v	54	w	55	x
56	y	57	z				

https://blog.boot.dev/bitcoin/base64-vs-base58-encoding/

- **Base64의 보완**
    * Base64 인코딩과 유사하지만, Base64에서 사용되는 **일부 문자('+', '/', '=')가 URL 등에서 문제를 일으키거나 쉽게 오해될 수 있어**, 이를 보완하기 위해 Base58이 등장했음

- **Base58의 용도**
    * Base58 인코딩은 주로 비트코인 및 암호화폐 주소
    * 가장 대표적인 사용처로, 비트코인(BTC)의 레거시(Legacy) 주소(1로 시작)와 같은 데이터를 짧고 입력하기 쉬운 문자열로 인코딩하는 데 사용
    * 개인 키(Private Key) 인코딩
        - WIF(Wallet Import Format) 형식의 개인 키를 인코딩할 때도 Base58이 사용

예시) BukQL을 Base58 디코딩 하면 10진수의 123456789.



```
# pip install base58

import base58

# 인코딩할 원본 문자열
original_text = "Hello Base58 World! "

# 1. 문자열을 바이트로 변환
data_to_encode = original_text.encode('utf-8')

# 2. Base58 인코딩
# 결과는 바이트 문자열 (b'...') 형태로 반환

encoded_bytes = base58.b58encode(data_to_encode)

# 3. Base58 인코딩된 바이트를 문자열로 변환하여 출력 (선택 사항)

encoded_string = encoded_bytes.decode('utf-8')
print(f"원본 문자열: {original_text}")
print(f"Base58 인코딩 (바이트): {encoded_bytes}")
print(f"Base58 인코딩 (문자열): {encoded_string}")

print("-" * 30)

# 4. Base58 디코딩

# 입력은 바이트 문자열 (encoded_bytes) 또는 문자열 (encoded_string) 모두 가능
decoded_bytes = base58.b58decode(encoded_bytes)
# decoded_bytes = base58.b58decode(encoded_string) # 이렇게도 가능

# 5. 디코딩된 바이트를 다시 원래 문자열로 변환
decoded_text = decoded_bytes.decode('utf-8')
```

```
## Base58Check 인코딩 및 디코딩

## 비트코인 주소 등에서는 오류 검출을 위해 4바이트의 체크섬(checksum)을
## 추가하는 Base58Check 인코딩 방식을 사용
## base58 라이브러리는 이 기능도 지원

import base58

data = b'base58 check test data'

# Base58Check 인코딩 (체크섬이 포함됩니다)
encoded_check = base58.b58encode_check(data)
print(f"원본 데이터: {data}")
print(f"Base58Check 인코딩: {encoded_check}")

# Base58Check 디코딩
try:
    decoded_check = base58.b58decode_check(encoded_check)
    print(f"Base58Check 디코딩: {decoded_check}")
    assert data == decoded_check
except ValueError as e:
    print(f"디코딩 오류: {e}")

print(f"Base58 디코딩 (바이트): {decoded_bytes}")
print(f"디코딩된 문자열: {decoded_text}")

# 결과 확인
assert original_text == decoded_text
```