



Chapter 2. 데이터의 표현

01 : 진법과 진법 변환

02 : 정수 표현

03 : 실수 표현

04 : 디지털 코드

05 : 에러 검출 코드



현대 컴퓨터의 부동소수점 수 표기

- IEEE 754 표기방식 사용
- Institute of Electrical and Electronics Engineers
- 단정도 부동소수점 수(32비트, float)
- 배정도 부동소수점 수(64비트, double)

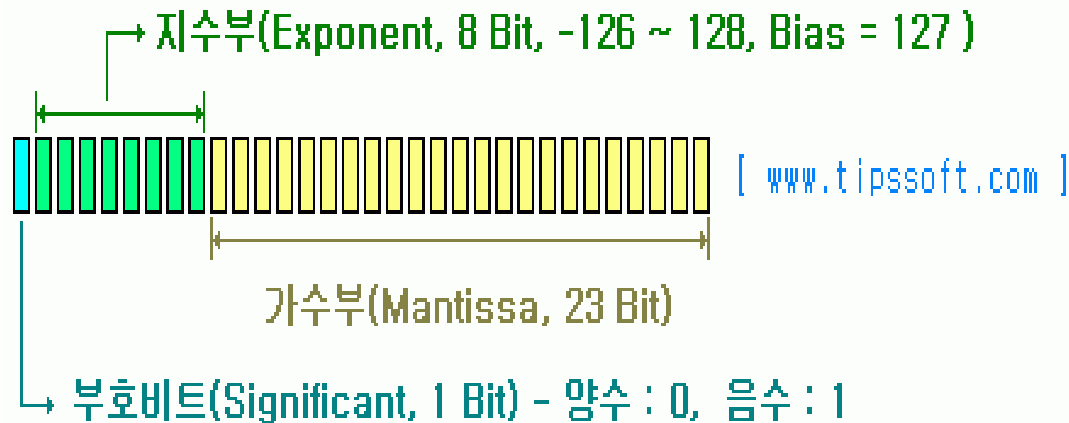
32비트 부동소수점 표현

1. 32비트 부동소수점 표현 - Single Precision (단정도, **float**)

[값의 저장 범위 : $1.2\text{E}-38 \sim 3.4\text{E}38$]

$$2^{128} = 3.4028236692093846346337460743177\text{e}+38$$

$$2^{-126} = 1.1754943508222875079687365372222\text{e}-38$$



0.4의 부동소수점 표현

□ 0.4의 2진수 표현 (부동 소수점)

$$0.0110011001100110011001100110 \times 2^0$$

1.1001 1001 1001 1001 1001 100 110 $\times 2^{-2}$ (정규화 된 수)

$-2 + 127 = 125 = 01111101_2$ **지수 -2를 +127로 바이어스**

부호 1비트, 지수부 8비트, 가수부 23비트

0 011 1110 1 100 1 100 1 100 1 100 1 100 1 101 <-반올림
 3 e c c c c c d

부호 (1비트)	지수부 (8비트)	가수부 (23비트)						
0	011 1110 1	1001	1001	1001	1001	1001	100	1 100
3	e	c	c	c	c	c	c → d	1에 의해 자리올림 발생

C프로그램으로 메모리에 존재하는 부동소수점 수를 확인하기

```
#include <stdio.h>

int main()
{
    float f = 43.25;
    printf("43.25 = %x\n", *(unsigned int *) &f);
    printf("43.25 = %a\n", f);

    getchar();
    return 0;
}
```

D:\work\float_test2015\Debug\float_test2015.exe

```
43.25 = 422d0000
43.25 = 0x1.5a0000p+5
```

64비트 부동소수점 표현

2. 64비트 부동소수점 표현 - Double Precision (배정도, double)

[값의 저장 범위 : $2.2\text{E}-308 \sim 1.8\text{E}308$]

$$2^{1024} = 1.797693134862315907729305190789\text{e}+308$$

$$2^{-1022} = 2.2250738585072013830902327173324\text{e}-308$$



Visual C 2010부동소수점 계산

```
#include <stdio.h>

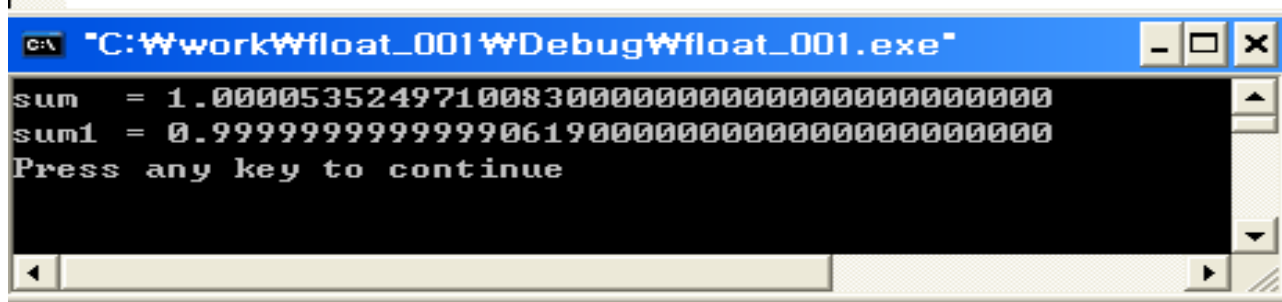
void main(void)
{
    int i;
    float sum=0.0;
    double sum1 = 0.0;

    for(i=0;i<10000;i++)
        sum += 0.0001;

    printf("sum  = %40.38f\\n",sum);

    for(i=0;i<10000;i++)
        sum1 += 0.0001;

    printf("sum1 = %40.38lf\\n",sum1);
}
```

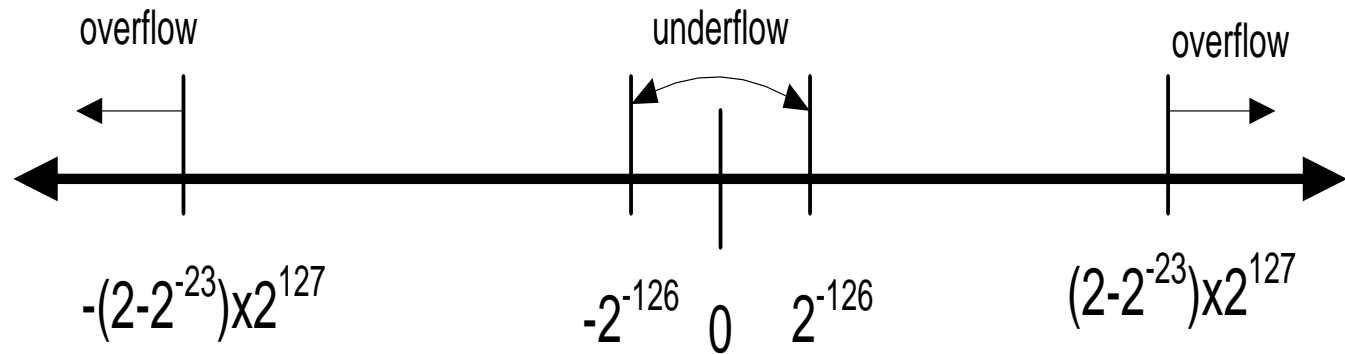


```
C:\work\float_001\Debug\float_001.exe
sum  = 1.000053524971008300000000000000000000000000
sum1 = 0.999999999999990619000000000000000000000000
Press any key to continue
```

0.0001을 10,000번 더했을 때의 결과가 왜 이렇게 나올까?

부동 소수점 수의 overflow, underflow

- overflow : 표현 범위를 넘어선 수
- underflow : 0에 무한히 가까운 수



단정도 부동 소수점 표현 범위

예제 2-3

1/256을 단정도 부동소수점 표현 방식으로 표현하시오.

$$0.00000001 = 1.0 \times 2^{-8} = 1/256$$

$$1/256 = 0.00390625$$

	0.00390625
×	2
	0.00781250
×	2
	0.0156250
×	2
	0.03125000
×	2
	0.06250000
×	2
	0.12500000
×	2
	0.25000000
×	2
	0.50000000
×	2
	1.00000000

예제 2-3

1/256을 단정도 부동소수점 표현 방식으로 표현하시오.

3b800000

0 011 1011 1 000 0000 0000 0000 0000 0000

양
수

119

1.00000000000000000000000000000000

-8 + 127 = 119 // 지수 -8에 바이어스 값 127을 더한다

$$1.0 \times 2^{-8} = 1/256$$

04 디지털 코드(Digital Code)

- BCD(Binary Coded Decimal, 8421) 코드
 - 10진수 0부터 9까지를 2진화한 코드로 실제 표기는 2진수이지만 10진수처럼 사용(4비트 코드로 1010 부터 1111은 사용안함)

BCD(8421) 코드 예

10진수	BCD 코드	10진수	BCD 코드	10진수	BCD 코드
0	0000	10	0001 0000	20	0010 0000
1	0001	11	0001 0001	31	0011 0001
2	0010	12	0001 0010	42	0100 0010
3	0011	13	0001 0011	53	0101 0011
4	0100	14	0001 0100	64	0110 0100
5	0101	15	0001 0101	75	0111 0101
6	0110	16	0001 0110	86	1000 0110
7	0111	17	0001 0111	97	1001 0111
8	1000	18	0001 1000	196	0001 1001 0110
9	1001	19	0001 1001	237	0010 0011 0111

BCD 코드 연산

6	0110	42	0100 0010
+ 3	+ 0011	+37	+0011 0111
----	-----	---	-----
9	1001	79	0111 1001
	1		2

8	1000	
+7	+ 0111	
---	-----	
	1111	결과가 9를 넘으면 6을 더함
	+ 0110	(+6)

15	0001 0101	3

3초과 코드

BCD 코드에 3(0011)을 더한 코드로 자기 보수의 성질을 가짐

현재 값에서 1의 보수를 취하면 10진수에서 9의 보수에 해당하는 값을 가짐

(10진수 6의 9의 보수는 3, 6의 3초과 코드는 1001이고 1의 보수는 0110)

10진 수	BCD 코드	3초과 코드
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

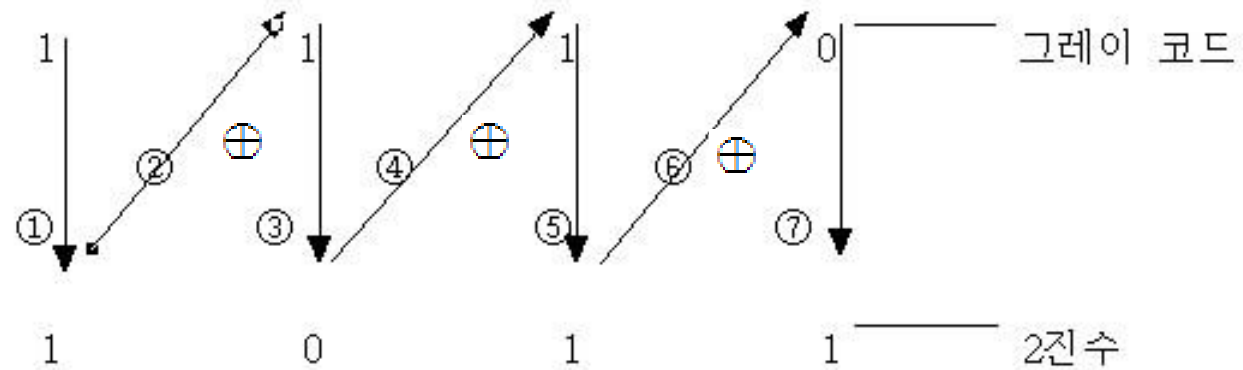
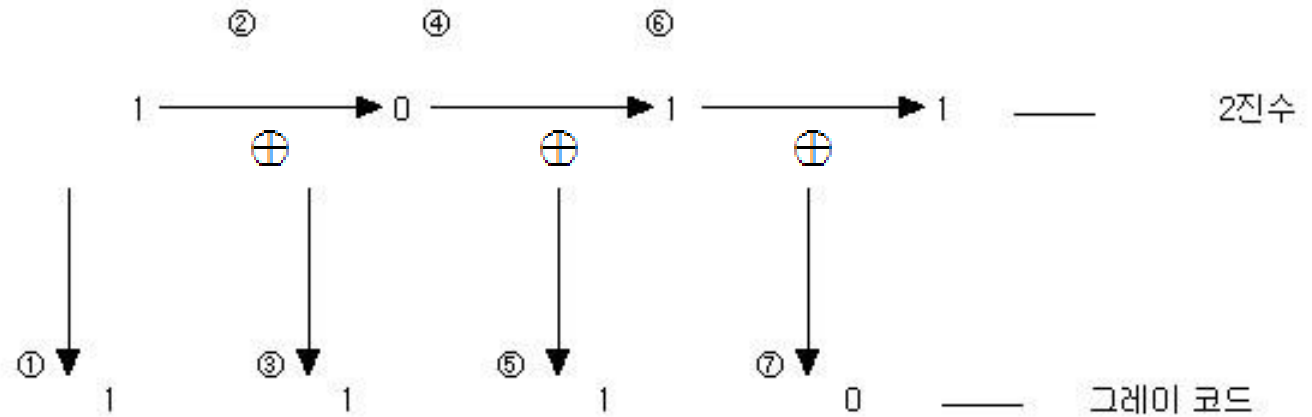
Gray code

- Gray Code : 인접한 10진 값 사이에 1비트만 변화

코드 10진수	2진수				그레이 코드			
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

<그레이코드와 2진수와의 비교표>

예를 들어, 2진수(1011)₂을 그레이 코드로 변환하는 과정은 다음과 같다.



다양한 2진 코드

가중치 코드

10진 수	8421 코드 (BCD)	2421 코드	5421 코드	84-2-1 코드	51111 코드	biquinary 코드	ring counter 9876543210
0	0000	0000	0000	0000	00000	0100001	0000000001
1	0001	0001	0001	0111	00001	0100010	0000000010
2	0010	0010	0010	0110	00011	0100100	0000000100
3	0011	0011	0011	0101	00111	0101000	0000001000
4	0100	0100	0100	0100	01111	0110000	0000010000
5	0101	1011	1000	1011	10000	1000001	0000100000
6	0110	1100	1001	1010	11000	1000010	0001000000
7	0111	1101	1010	1001	11100	1000100	0010000000
8	1000	1110	1011	1000	11110	1001000	0100000000
9	1001	1111	1100	1111	11111	1010000	1000000000

비가중치 코드

10진 수	excess-3 code	5중2코드	시프트카운터	그레이코드
0	0011	11000	00000	0000
1	0100	00011	00001	0001
2	0101	00101	00011	0011
3	0110	00110	00111	0010
4	0111	01001	01111	0110
5	1000	01010	11111	0111
6	1001	01100	11110	0101
7	1010	10001	11100	0100
8	1011	10010	11000	1100
9	1100	10100	10000	1101

영숫자 코드(alphanumeric code)

표준 BCD 코드(1비트 패리티, 6비트 정보비트)

패리티 비트	존 비트		숫자 비트			
6	5	4	3	2	1	0
하위 비트에 따라 달라짐	1	1	영문자 A~I(0001~1001)			
	1	0	영문자 J~R(0001~1001)			
	0	1	영문자 S~Z(0010~1001)			
	0	0	숫자 0~9(0001~1010)			
	혼용		특수 문자 및 기타 문자			

문자	P ZZ8421	문자	P ZZ8421	문자	P ZZ8421	문자	P ZZ8421	문자	P ZZ8421
A	0 110001	J	1 100001	S	1 010010	1	0 000001	=	0 001011
B	0 110010	K	1 100010	T	0 010011	2	0 000010	>	1 001100
C	1 110011	L	0 100011	U	1 010100	3	1 000011	+	0 010000
D	0 110100	M	1 100100	V	0 010101	4	0 000100	,	1 011011
E	1 110101	N	0 100101	W	0 010110	5	1 000101)	0 011100
F	1 110110	O	0 100110	X	1 010111	6	1 000110	%	1 011101
G	0 110111	P	1 100111	Y	1 011000	7	0 000111	?	0 011111
H	0 111000	Q	1 101000	Z	0 011001	8	0 001000	-	1 100001
I	1 111001	R	0 101001			9	1 001001	@	1 111010
						0	1 001010	\$	1 111111

ASCII코드

ASCII(American Standard Code for Information Interchange) code

- 미국 국립표준연구소가 제정한 정보교환용 미국 표준 코드
- 3비트 존과 4비트 디지트에 1비트의 패리티 비트를 추가하여 만든 8비트 코드(0~127의 128가지 문자를 표현)

USASCII code chart

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

b₈ : parity bit

b₇b₆b₅ : zone bit

b₄b₃b₂b₁ : digit bit

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

확장된 ASCII Code : IBM System/360에 처음으로 사용, 주로 대형컴퓨터와 IBM에서 사용

Extended ASCII Codes

As people gradually required computers to understand additional characters and non-printing characters the ASCII set became restrictive. As with most technology, it took a while to get a single standard for these extra characters and hence there are few varying 'extended' sets. The most popular is presented below.

128	Ç	144	É	160	á	176	░	193	⌞	209	〒	225	β	241	±
129	ù	145	æ	161	í	177	▒	194	⌟	210	⌠	226	Γ	242	≥
130	é	146	Æ	162	ó	178	▓	195	⌡	211	⌢	227	π	243	≤
131	â	147	ô	163	ú	179		196	—	212	⌣	228	Σ	244	∫
132	ä	148	ö	164	ñ	180	⌤	197	⌥	213	⌦	229	σ	245	∫
133	à	149	ò	165	Ñ	181	⌥	198	⌧	214	⌨	230	μ	246	÷
134	å	150	û	166	ª	182	⌦	199	〈	215	〉	231	τ	247	≈
135	ç	151	ù	167	º	183	⌧	200	〉	216	⌫	232	Φ	248	°
136	ê	152	—	168	¿	184	⌨	201	⌭	217	⌬	233	⊕	249	·
137	ë	153	Ö	169	—	185	〈	202	⌮	218	⌭	234	Ω	250	·
138	è	154	Ü	170	¬	186	〉	203	⌯	219	■	235	δ	251	√
139	ï	156	£	171	½	187	⌭	204	⌰	220	■	236	∞	252	—
140	î	157	¥	172	¼	188	⌮	205	=	221	■	237	φ	253	²
141	ì	158	—	173	¡	189	⌮	206	⌱	222	■	238	ε	254	■
142	Ä	159	ƒ	174	«	190	⌰	207	⌲	223	■	239	∩	255	
143	Å	192	Ł	175	»	191	⌱	208	⌳	224	α	240	≡		

Source: www.asciitable.com

- EBCDIC(Extended Binary Coded Decimal Interchange code)
- IBM의 System/360에 처음사용, 대형컴퓨터와 IBM 계열 사용

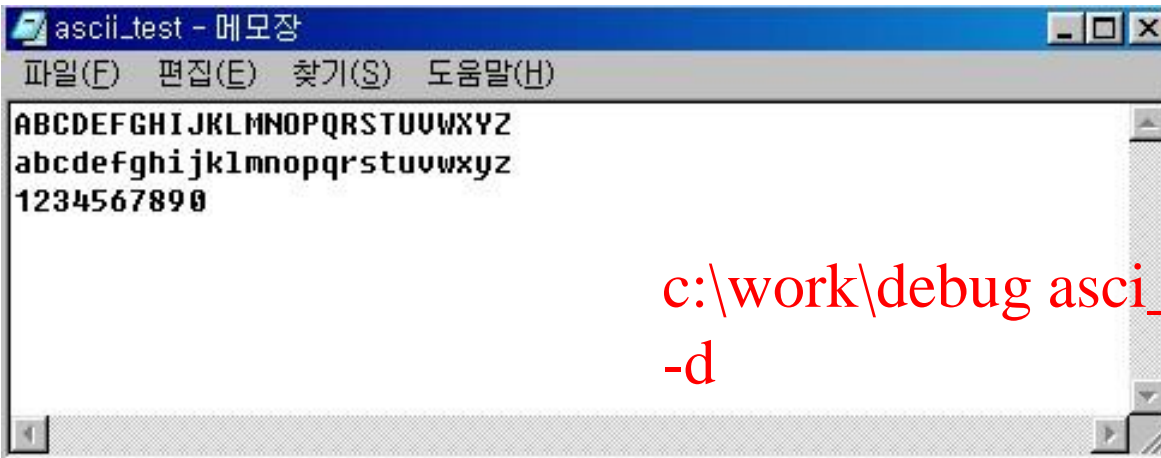
EBCDIC Codes

ASCII is not the only format in use out there. IBM adopted EBCDIC (Extended Binary Coded Decimal Interchange Code) developed for punched cards in the early 1960s and still uses it on mainframes today. It is probably the next most well known character set due to the proliferation of IBM mainframes. It comes in at least six slightly differing forms, so again here is the most common.

Dec	Hx	Oct	Char		Dec	Hx	Oct	Char		Dec	Hx	Oct	Char		Dec	Hx	Oct	Char
0	0	000	nul	(Null)	65	41	101			130	82	202	b		195	c3	303	C
1	1	001	soh	(Start of Heading)	66	42	102			131	83	203	c		196	c4	304	D
2	2	002	stx	(Start of Text)	67	43	103			132	84	204	d		197	c5	305	E
3	3	003	etx	(End of Text)	68	44	104			133	85	205	e		198	c6	306	F
4	4	004	pf	(Punch Off)	69	45	105			134	86	206	f		199	c7	307	G
5	5	005	ht	(Horizontal Tab)	70	46	106			135	87	207	g		200	c8	310	H
6	6	006	lc	(Lower Case)	71	47	107			136	88	210	h		201	c9	311	I
7	7	007	del	(Delete)	72	48	110			137	89	211	i		202	ca	312	
8	8	010	ge		73	49	111			138	8a	212			203	cb	313	
9	9	011	rlf		74	4a	112	ø		139	8b	213			204	cc	314	
10	a	012	smm	(Start of Manual Message)	75	4b	113	.		140	8c	214			205	cd	315	
11	b	013	vt	(Vertical Tab)	76	4c	114	>		141	8d	215			206	ce	316	
12	c	014	ff	(Form Feed)	77	4d	115	(142	8e	216			207	cf	317	
13	d	015	cr	(Carriage Return)	78	4e	116	+		143	8f	217			208	d0	320	}
14	e	016	so	(Shift Out)	79	4f	117			144	90	220			209	d1	321	J
15	f	017	si	(Shift in)	80	50	120	&		145	91	221	j		210	d2	322	K
16	10	020	dle	(Data Link Escape)	81	51	121			146	92	222	k		211	d3	323	L
17	11	021	dc1	(Device Control 1)	82	52	122			147	93	223	l		212	d4	324	M
18	12	022	dc2	dc2 (Device Control 2)	83	53	123			148	94	224	m		213	d5	325	N
19	13	023	tm	(Tape Mark)	84	54	124			149	95	225	n		214	d6	326	O
20	14	024	res	(Restore)	85	55	125			150	96	226	o		215	d7	327	P
21	15	025	nl	(New Line)	86	56	126			151	97	227	p		216	d8	330	Q
22	16	026	bs	(Backspace)	87	57	127			152	98	230	q		217	d9	331	R
23	17	027	il	(Idle)	88	58	130			153	99	231	r		218	da	332	
24	18	030	can	(Cancel)	89	59	131			154	9a	232			219	db	333	
25	19	031	em	(End of Medium)	90	5a	132	!		155	9b	233			220	dc	334	
26	1a	032	cc	(Cursor Control)	91	5b	133	\$		156	9c	234			221	dd	335	
27	1b	033	cu1	(Customer Use 1)	92	5c	134	*		157	9d	235			222	de	336	
28	1c	034	ifs	(Interchange File Separator)	93	5d	135)		158	9e	236			223	df	337	
29	1d	035	igs	(Interchange Group Separator)	94	5e	136	;		159	9f	237			224	e0	340	\
30	1e	036	irs	(Interchange Record)	95	5f	137			160	a0	240			225	e1	341	
31	1f	037	ius	(Interchange Unit Separator)	96	60	140	-		161	a1	241	~		226	e2	342	S
32	20	040	ds	(Digit Select)	97	61	141	/		162	a2	242	s		227	e3	343	T
33	21	041	sos	(Start of Significance)	98	62	142			163	a3	243	t		228	e4	344	U
34	22	042	fs	(Field Separator)	99	63	143			164	a4	244	u		229	e5	345	V
35	23	043			100	64	144			165	a5	245	v		230	e6	346	W
36	24	044	byp	(Bypass)	101	65	145			166	a6	246	w		231	e7	347	X

ASCII code의 사용확인(debug 사용해보기)

debug : windows 7 32비트 이하에서 사용가능



c:\work\debug ascii_test.txt
-d

```
C:\wythyun\2003_강의자료\컴퓨터구조_2\chapter_3>debug ascii_~1.txt
```

```
-d
```

171E:0100	41	42	43	44	45	46	47	48-49	4A	4B	4C	4D	4E	4F	50	ABCDEFGHIJKLMNOP
171E:0110	51	52	53	54	55	56	57	58-59	5A	0D	0A	61	62	63	64	QRSTUVWXYZ..abcd
171E:0120	65	66	67	68	69	6A	6B	6C-6D	6E	6F	70	71	72	73	74	efghijklmnopqrst
171E:0130	75	76	77	78	79	7A	0D	0A-31	32	33	34	35	36	37	38	uvwxyz..12345678
171E:0140	39	30	0D	0A	E7	1F	89	36-F3	DF	8A	D8	33	C0	81	FF	90.....6....3...
171E:0150	DE	E9	74	21	A0	F5	DF	0A-C9	75	05	3A	45	FF	74	01	..t!.....u.:E.t.
171E:0160	AA	8B	F2	AC	AA	0A	C0	75-FA	BA	DE	E9	C7	06	45	E0u.....E.
171E:0170	F6	84	E8	10	00	0A	DB	74-04	9D	F8	EB	02	9D	F9	5Et.....^

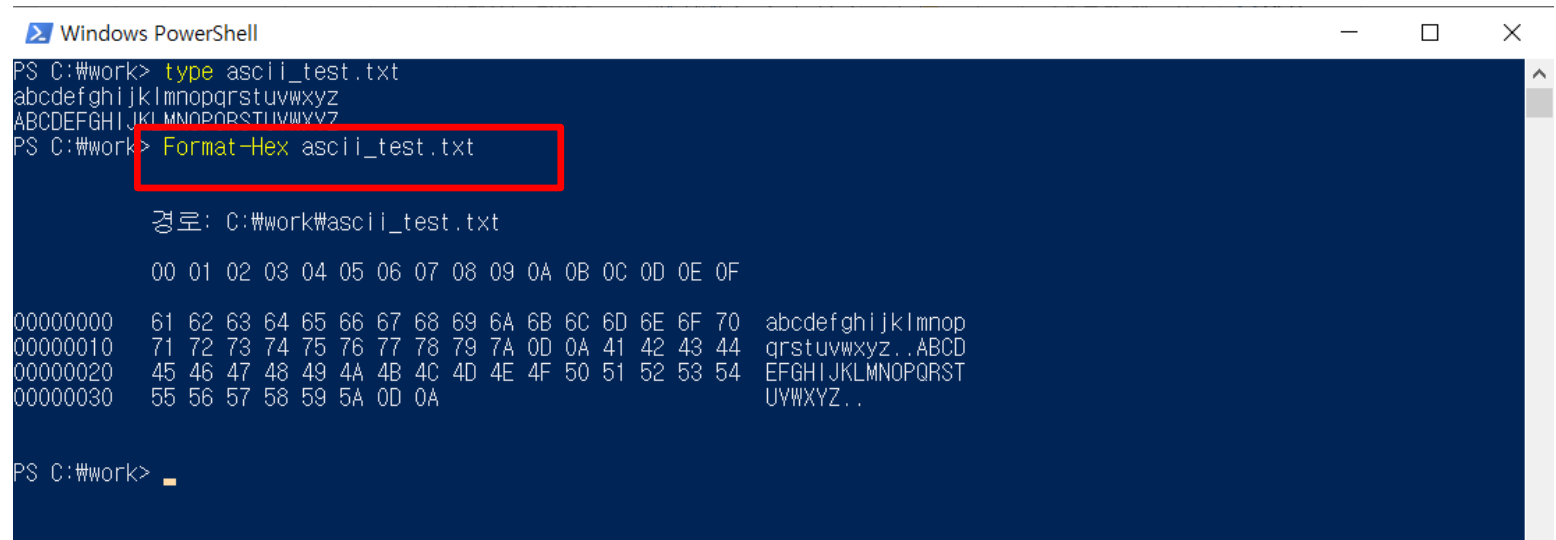
ASCII code의 사용확인(Format-Hex사용해보기)

Format-Hex : windows10 PowerShell에서 사용가능



```
ascii_test - 메모장
파일(F) 편집(E) 찾기(S) 도움말(H)
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890
```

c:\work\Format-Hex ascii_test.txt



```
Windows PowerShell
PS C:\work> type ascii_test.txt
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
PS C:\work> Format-Hex ascii_test.txt

경로: C:\work\ascii_test.txt

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 abcdefghijklmnop
00000010 71 72 73 74 75 76 77 78 79 7A 0D 0A 41 42 43 44 qrstuvwxyz..ABCD
00000020 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 EFGHIJKLMNOPQRST
00000030 55 56 57 58 59 5A 0D 0A UVWXYZ..

PS C:\work>
```


확장된 ASCII code C code로 출력하기

```
#include <stdio.h>
#include <stdlib.h>

void main(void)
{
    int i;

    system("chcp 437"); //MS-DOS 영문모드 설정

    for(i=32;i<256;i++) {
        printf("%c(%03d) ",i, i);
        if(((i+1)%8) == 0) //8 문자마다 줄 바뀜 표시
            printf("\n");
    }
    getchar();
}
```

경우에 따라 이프로그램 코드를 윈도우10에서
실행할 때 NTVDM관련 창이뜨면 설치해 주시면 됩니다.
인터넷에 NTVDM 찾아보시면 기능을 알 수 있습니다.

출력 결과

```
C:\Users\dockotac\source\repos\Project26\Debug\Project26.exe
Active code page: 437
(00032) ! (00033) " (00034) # (00035) $ (00036) % (00037) & (00038) ' (00039)
(00040) ) (00041) * (00042) + (00043) , (00044) - (00045) . (00046) / (00047)
0 (00048) 1 (00049) 2 (00050) 3 (00051) 4 (00052) 5 (00053) 6 (00054) 7 (00055)
8 (00056) 9 (00057) : (00058) ; (00059) < (00060) = (00061) > (00062) ? (00063)
@ (00064) A (00065) B (00066) C (00067) D (00068) E (00069) F (00070) G (00071)
H (00072) I (00073) J (00074) K (00075) L (00076) M (00077) N (00078) O (00079)
P (00080) Q (00081) R (00082) S (00083) T (00084) U (00085) V (00086) W (00087)
X (00088) Y (00089) Z (00090) [ (00091) \ (00092) ] (00093) ^ (00094) _ (00095)
(00096) a (00097) b (00098) c (00099) d (00100) e (00101) f (00102) g (00103)
h (00104) i (00105) j (00106) k (00107) l (00108) m (00109) n (00110) o (00111)
p (00112) q (00113) r (00114) s (00115) t (00116) u (00117) v (00118) w (00119)
x (00120) y (00121) z (00122) { (00123) | (00124) } (00125) ~ (00126) ¨ (00127)
C (00128) ü (00129) é (00130) â (00131) ä (00132) à (00133) å (00134) ç (00135)
è (00136) ë (00137) è (00138) ï (00139) î (00140) ï (00141) Ä (00142) Å (00143)
É (00144) æ (00145) Æ (00146) ô (00147) ö (00148) ò (00149) ù (00150) ù (00151)
ÿ (00152) ö (00153) ü (00154) ø (00155) € (00156) ¥ (00157) ₣ (00158) ¢ (00159)
à (00160) í (00161) ó (00162) ú (00163) ñ (00164) Ñ (00165) ª (00166) º (00167)
¿ (00168) ¸ (00169) ¬ (00170) ½ (00171) ¼ (00172) j (00173) « (00174) » (00175)
(00176) ▯ (00177) ▯ (00178) | (00179) + (00180) + (00181) + (00182) + (00183)
(00184) + (00185) || (00186) ¶ (00187) ¶ (00188) ¶ (00189) ¶ (00190) ¶ (00191)
(00192) + (00193) + (00194) + (00195) + (00196) + (00197) + (00198) + (00199)
(00200) ¶ (00201) ¶ (00202) ¶ (00203) ¶ (00204) = (00205) + (00206) = (00207)
(00208) ¶ (00209) ¶ (00210) ¶ (00211) ¶ (00212) ¶ (00213) ¶ (00214) + (00215)
(00216) + (00217) ¶ (00218) ▯ (00219) ▯ (00220) ▯ (00221) ▯ (00222) ▯ (00223)
α (00224) β (00225) γ (00226) π (00227) Σ (00228) σ (00229) μ (00230) τ (00231)
Φ (00232) Θ (00233) Ω (00234) δ (00235) ∞ (00236) φ (00237) ε (00238) ∩ (00239)
≡ (00240) ± (00241) ≥ (00242) ≤ (00243) | (00244) | (00245) ÷ (00246) ≈ (00247)
° (00248) · (00249) · (00250) √ (00251) ⁿ (00252) ⁻ (00253) ■ (00254) (00255)
```



Unicode(유니코드)

- 플랫폼, 프로그램, 언어에 상관없이 모든 문자에 고유번호 제공
- 인터넷 시대의 표준코드
- 유니코드 컨소시엄은 32비트(UTF-32), 16비트(UTF-16), 8비트(UTF-8) 세 가지 코드 제안
- XML, 자바, ECMAScript(자바스크립트), LDAP, CORBA 3.0, WML 등 현재 사용되는 표준에 필요, ISO/IEC 10646 구현

기출문제 풀이

- 16진수 C52를 2진수로 변환하시오
- 다음 진수 표현 중 가장 큰 수는?
1) FF_{16} 2) 257_{10} 3) 11111111_2 4) 377_8



수고하셨습니다!