

데이터와 메모리 (Data and memory)



주식회사 가치랩스

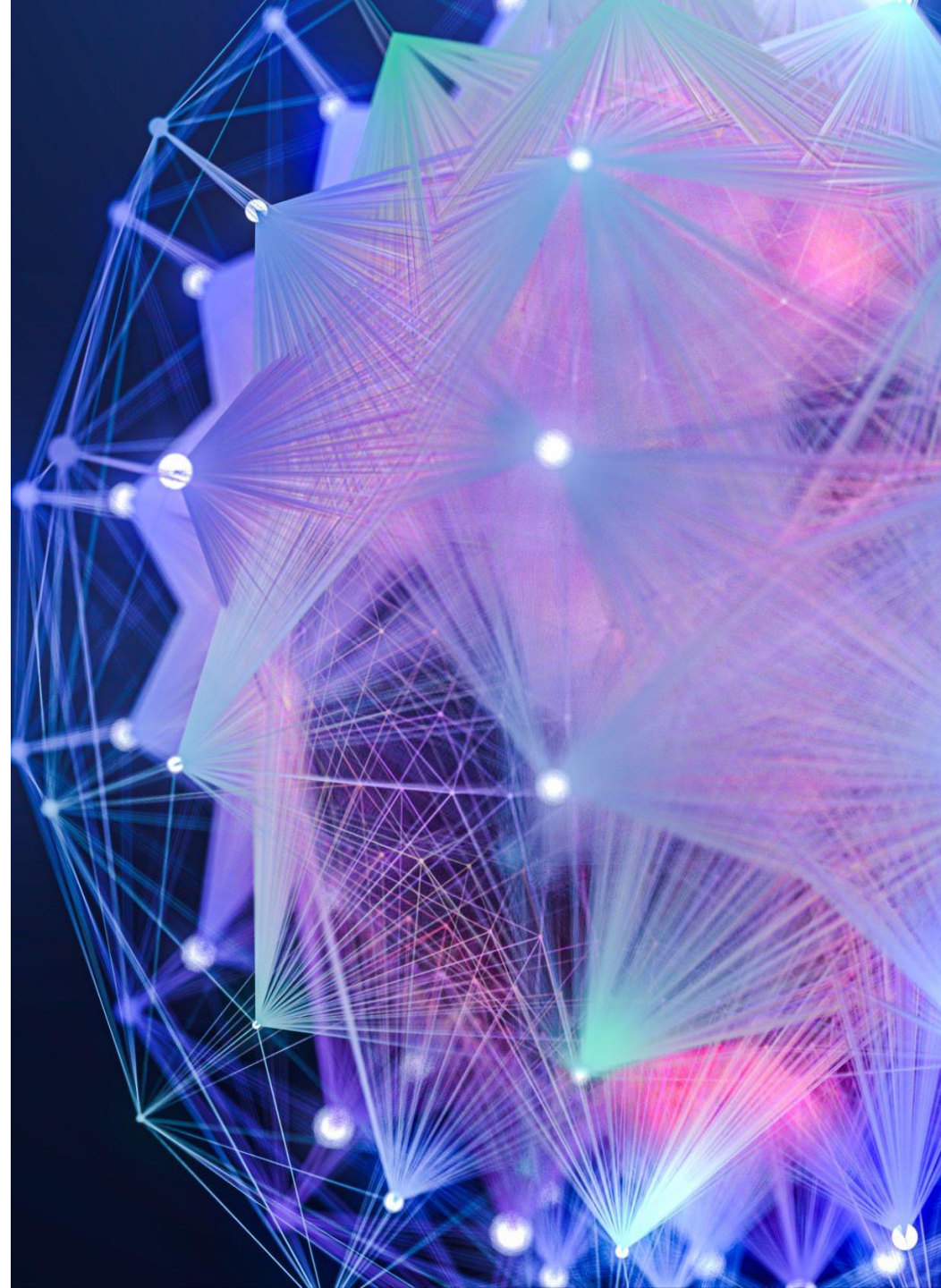
GazziLabs, Inc.

<http://gazzi.ai>

안기옥

kiokahn@gazzi.ai

<https://github.com/kiokahn/DataAndMemory>



전체 목차

1. 데이터와 메모리
2. C++에서 변수 선언과 메모리 할당
3. 데이터와 메모리,파일 연습



1. 데이터와 메모리

- 개요
- 시스템과 메모리
- 메모리에 저장되는 데이터 - 문자
- 메모리에 저장되는 데이터 - 소리(음악)
- 메모리에 저장되는 데이터 - 영상
- 메모리에 저장된 값에 대한 연산

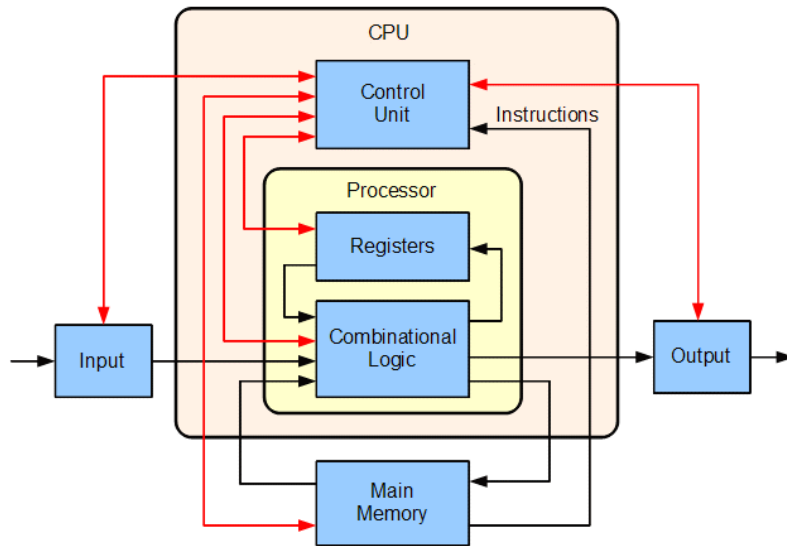


❖ 컴퓨터 종류



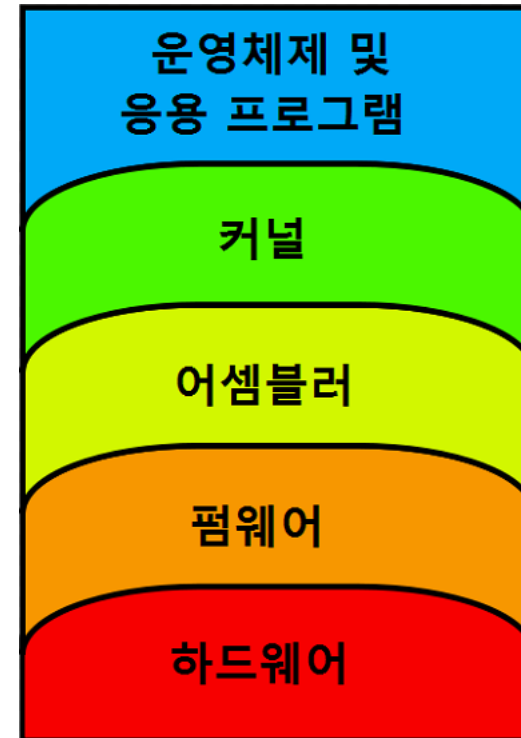
❖ 폰노이만 컴퓨터


- EDVAC에 대한 보고서의 첫 번째 초안
(First Draft of a Report on the EDVAC), 1945



※ 참고 : https://ko.wikipedia.org/wiki/컴퓨터_구조

❖ 컴퓨터 계층구조



 releases

Ubuntu 16.04.7 LTS (Xenial Xerus)

Select an image

Ubuntu is distributed on two types of images described below.

Desktop image

The desktop image allows you to try Ubuntu without changing your computer at all, and at your option to install it permanently later. This type of image is what most people will want to use. You will need at least 384MiB of RAM to install from this image.

64-bit PC (AMD64) desktop image

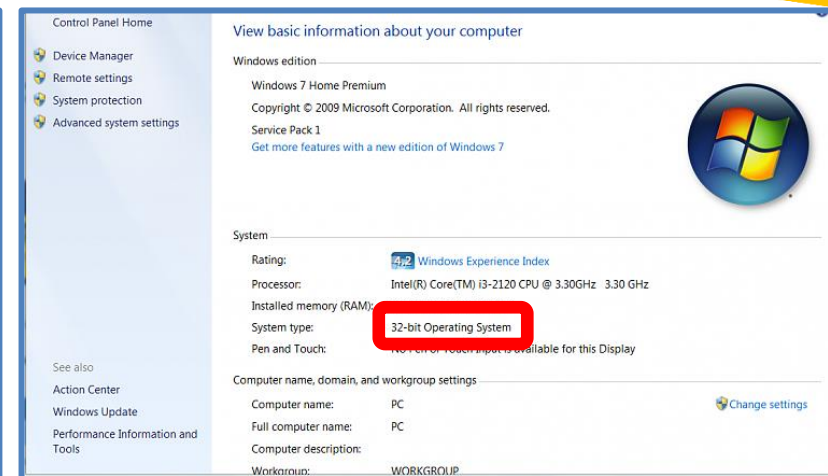
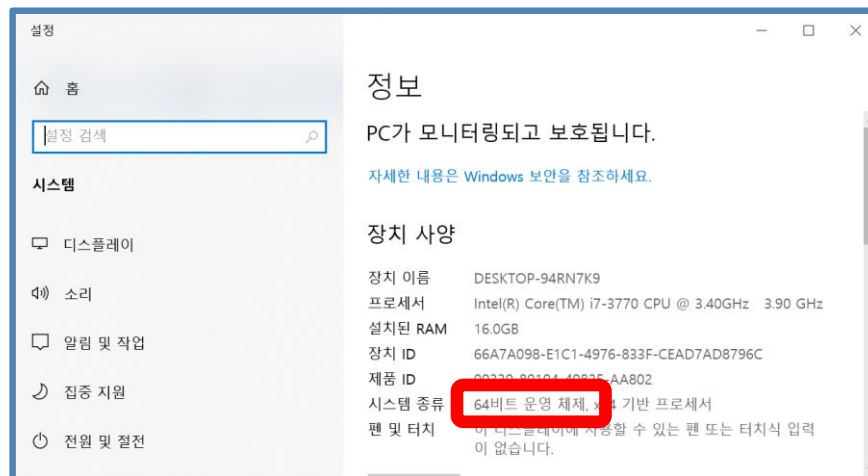
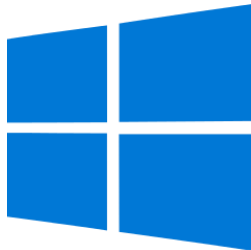
Choose this if you have a computer based on the AMD64 or EM64T architecture (e.g., Athlon64, Opteron, EM64T Xeon, Core 2). Choose this if you are at all unsure.

32-bit PC (i386) desktop image

For almost all PCs. This includes most machines with Intel/AMD/etc type processors and almost all computers that run Microsoft Windows, as well as newer Apple Macintosh systems based on Intel processors.

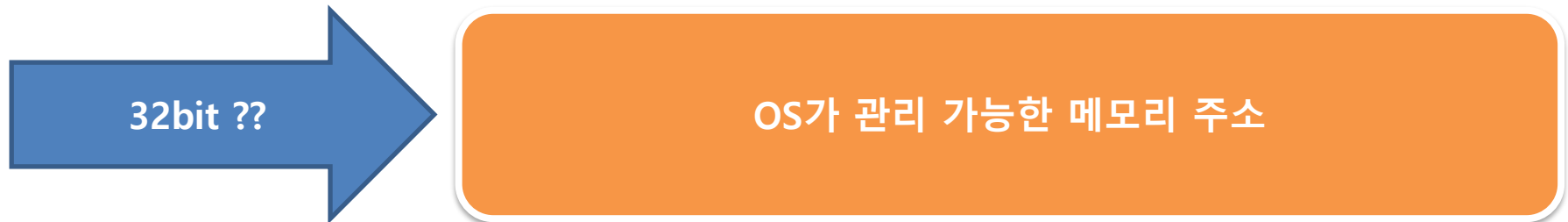
<https://releases.ubuntu.com/16.04/>

개요 - OS (Operating System), MS Windows



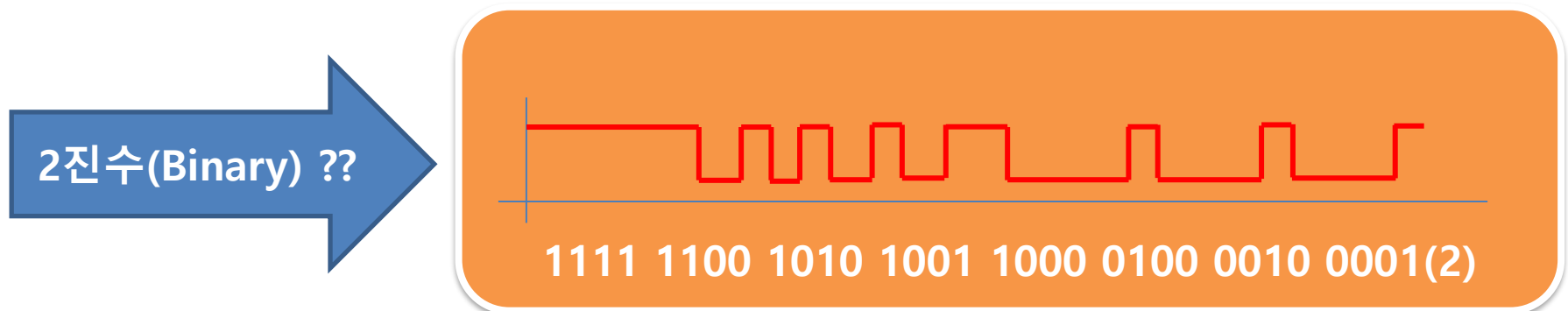
❖ 내 컴퓨터는 32bit시스템 입니다.

- 내 컴퓨터의 운영체제(OS, Operating System)는 32bit 입니다.



❖ **Bit** (0 or 1) : 컴퓨터는 2진수(Binary)를 사용 합니다.

- 컴퓨터는 구조상 2진 데이터만 운용 가능합니다.



❖ 진수

- 1010(2) → Binary
- 10 → Decimal
- 0xA → Hexadecimal

❖ Hex

- ^{8 4 2 1} 1010(2)=10=0xA, 11=0xB, 12=0xC, 13=0xD, 14=0xE, 15=0xF

❖ Bin To Dec

$$\begin{aligned} 1111(2) &= 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\ &= 8 + 4 + 2 + 1 \\ &= 15 \end{aligned}$$

$$\begin{aligned} 1111 \ 1111(2) &= 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\ &= 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 \\ &= 255 \end{aligned}$$

❖ Bin To Hex

- 1111 1111(2) = 255 = 0xFF

16 | 255

15...15
0x. F. F

- ^{8 4 2 1 8 4 2 1} 1111 1100 1010 1001 1000 0100 0010 0001(2) = 0xFCA9 8421

_{= 15 = 12}
_{= 0xF = 0xC}

→ 2Byte → 2Byte

MSB : Most Significant Bit

LSB : Least Significant Bit

4개씩 나누면,
쉽다

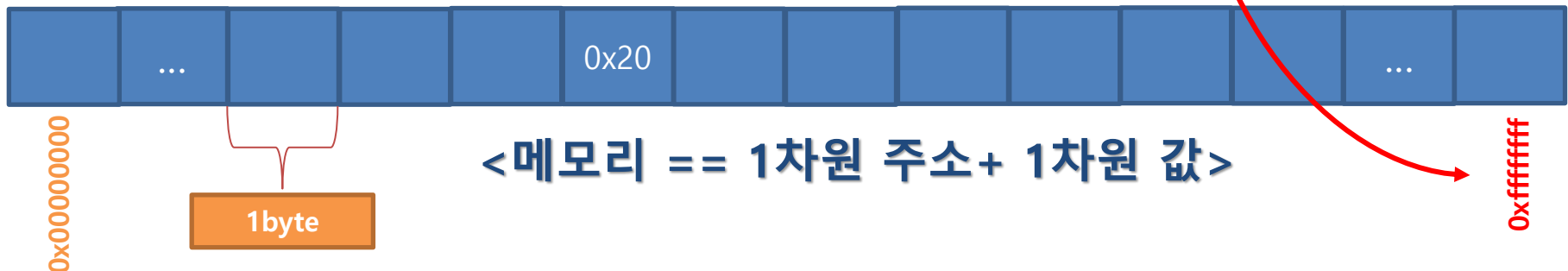
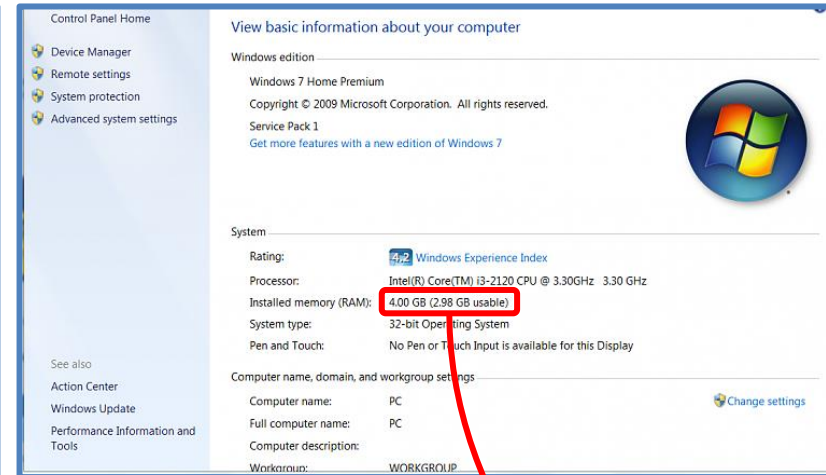
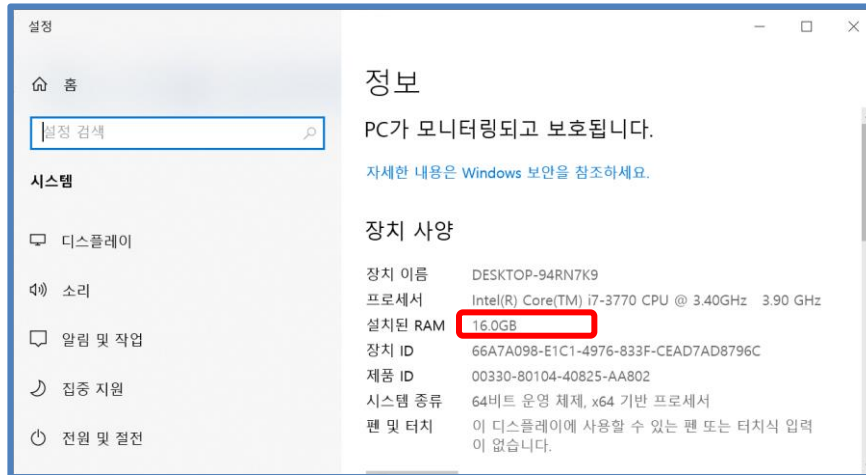
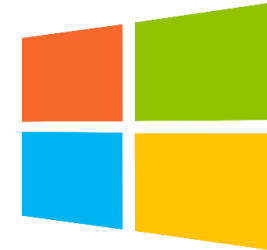
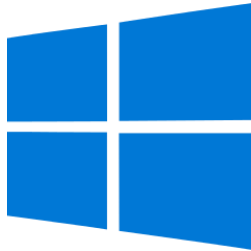


메모리 관리범위 = 0x00000000번지~0xFFFFFFFF 번지
(1byte 단위로 주소 관리)

최대 크기 = 0xFFFFFFFF + 1 Byte
= 4,294,967,295 + 1 byte
= 4,294,967,296 Byte
= 4,194,304 Kbyte (/1024)
= 4,096 Mbyte (/1024)
= 4 Gbyte

- ❖ 32bit(4byte) OS에서 관리 가능한 최대 메모리는 4G byte 입니다.
 - ❖ 32bit 변수로 관리 가능한 최대 파일 크기는 4G byte 입니다.
 - ❖ 64bit OS의 경우,
 - 64bit 주소는 8 Bytes
 - 최대 메모리 주소 번지 == 0xFFFFFFFF FFFFFFFF
 - 16EB(엑사바이트, M→G→T→P→E) → 현재는 지원 가능한 OS 없음
- Windows 10 Enterprise : 2TB
Windows 10 Home : 128GB

시스템과 메모리 – 메모리(RAM) 크기

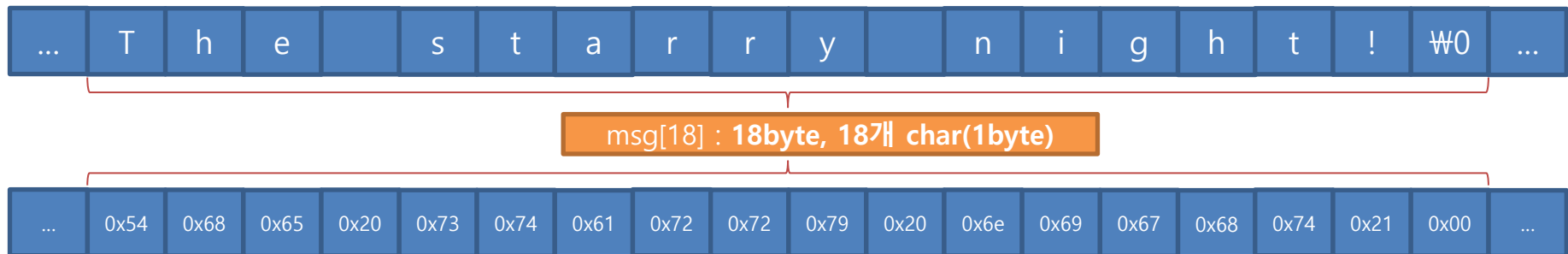


❖ 변수형 : char

```
#include <iostream>

using namespace std;

int main(void)
{
    char msg[18] = "The starry night!";
    cout << msg << endl;
    return 0;
}
```



문자의 숫자화 ? → **ASCII Code**(American Standard Code for Information Interchange, 1963)

※ 참고 : <https://www.techtarget.com/whatis/definition/ASCII-American-Standard-Code-for-Information-Interchange>

메모리에 저장되는 데이터 - 문자 ASCII Code

10진수	2진수	8진수	16진수	약자
0	00000000	000	0x00	NUL
1	00000001	001	0x01	SOH
2	00000010	002	0x02	STX
3	00000011	003	0x03	ETX
4	00000100	004	0x04	EOT
5	00000101	005	0x05	ENQ
6	00000110	006	0x06	ACK
7	00000111	007	0x07	BEL
8	00001000	010	0x08	BS
9	00001001	011	0x09	HT
10	00001010	012	0x0A	LF
11	00001011	013	0x0B	VT
12	00001100	014	0x0C	FF
13	00001101	015	0x0D	CR
14	00001110	016	0x0E	SO
15	00001111	017	0x0F	SI
16	00010000	020	0x10	DLE
17	00010001	021	0x11	DC1
18	00010010	022	0x12	DC2
19	00010011	023	0x13	DC3
20	00010100	024	0x14	DC4
21	00010101	025	0x15	NAK
22	00010110	026	0x16	SYN
23	00010111	027	0x17	ETB
24	00011000	030	0x18	CAN
25	00011001	031	0x19	EM
26	00011010	032	0x1A	SUB
27	00011011	033	0x1B	ESC
28	00011100	034	0x1C	FS
29	00011101	035	0x1D	GS
30	00011110	036	0x1E	RS
31	00011111	037	0x1F	US

10진수	2진수	8진수	16진수	문자
32	00100000	040	0x20	
33	00100001	041	0x21	!
34	00100010	042	0x22	"
35	00100011	043	0x23	#
36	00100100	044	0x24	\$
37	00100101	045	0x25	%
38	00100110	046	0x26	&
39	00100111	047	0x27	'
40	00101000	050	0x28	(
41	00101001	051	0x29)
42	00101010	052	0x2A	*
43	00101011	053	0x2B	+
44	00101100	054	0x2C	,
45	00101101	055	0x2D	-
46	00101110	056	0x2E	.
47	00101111	057	0x2F	/
48	00110000	060	0x30	0
49	00110001	061	0x31	1
50	00110010	062	0x32	2
51	00110011	063	0x33	3
52	00110100	064	0x34	4
53	00110101	065	0x35	5
54	00110110	066	0x36	6
55	00110111	067	0x37	7
56	00111000	070	0x38	8
57	00111001	071	0x39	9
58	00111010	072	0x3A	:
59	00111011	073	0x3B	;
60	00111100	074	0x3C	<
61	00111101	075	0x3D	=
62	00111110	076	0x3E	>
63	00111111	077	0x3F	?
64	01000000	100	0x40	@

10진수	2진수	8진수	16진수	문자
65	01000001	101	0x41	A
66	01000010	102	0x42	B
67	01000011	103	0x43	C
68	01000100	104	0x44	D
69	01000101	105	0x45	E
70	01000110	106	0x46	F
71	01000111	107	0x47	G
72	01001000	110	0x48	H
73	01001001	111	0x49	I
74	01001010	112	0x4A	J
75	01001011	113	0x4B	K
76	01001100	114	0x4C	L
77	01001101	115	0x4D	M
78	01001110	116	0x4E	N
79	01001111	117	0x4F	O
80	01010000	120	0x50	P
81	01010001	121	0x51	Q
82	01010010	122	0x52	R
83	01010011	123	0x53	S
84	01010100	124	0x54	T
85	01010101	125	0x55	U
86	01010110	126	0x56	V
87	01010111	127	0x57	W
88	01011000	130	0x58	X
89	01011001	131	0x59	Y
90	01011010	132	0x5A	Z
91	01011011	133	0x5B	[
92	01011100	134	0x5C	\
93	01011101	135	0x5D]
94	01011110	136	0x5E	^
95	01011111	137	0x5F	_
96	01100000	140	0x60	`

10진수	2진수	8진수	16진수	문자
97	01100001	141	0x61	a
98	01100010	142	0x62	b
99	01100011	143	0x63	c
100	01100100	144	0x64	d
101	01100101	145	0x65	e
102	01100110	146	0x66	f
103	01100111	147	0x67	g
104	01101000	150	0x68	h
105	01101001	151	0x69	i
106	01101010	152	0x6A	j
107	01101011	153	0x6B	k
108	01101100	154	0x6C	l
109	01101101	155	0x6D	m
110	01101110	156	0x6E	n
111	01101111	157	0x6F	o
112	01110000	160	0x70	p
113	01110001	161	0x71	q
114	01110010	162	0x72	r
115	01110011	163	0x73	s
116	01110100	164	0x74	t
117	01110101	165	0x75	u
118	01110110	166	0x76	v
119	01110111	167	0x77	w
120	01111000	170	0x78	x
121	01111001	171	0x79	y
122	01111010	172	0x7A	z
123	01111011	173	0x7B	{
124	01111100	174	0x7C	
125	01111101	175	0x7D	}
126	01111110	176	0x7E	~
127	01111111	177	0x7F	DEL

메모리에 저장되는 데이터 – 소리(음악)

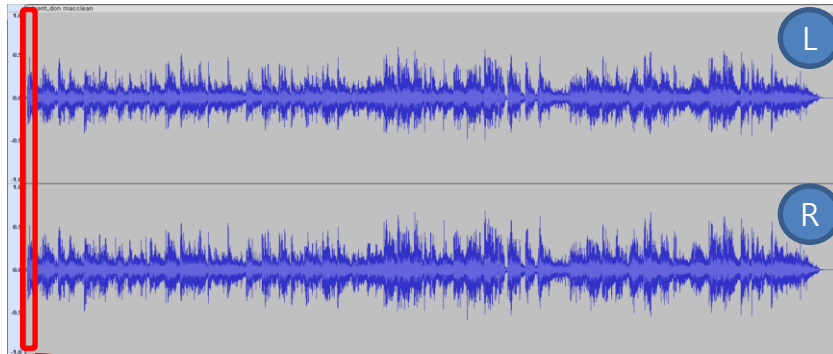
❖ 변수형 : short



vicent_don_macclean.wav : ※참고 : <https://sourceforge.net/projects/mediainfo/>

<MediaInfo>

Format	: PCM
Format settings	: Little / Signed
Codec ID	: 1
Duration	: 4 min 3 s
Bit rate mode	: Constant
Bit rate	: 1 411.2 kb/s
Channel(s)	: 2 channels → Left, Right
Sampling rate	: 44.1 kHz → 44,100개/1초, 44.1개/0.001초
Bit depth	: 16 bits → 2bytes 변수 : short
Stream size	: 41.0 MiB (100%)



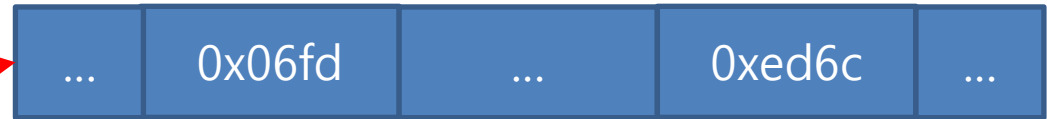
32,767
(0x7FFF)

-32,768
(0x8000)

음수 : 2의 보수

0.0847s 구간, 0.001초

176byte, 44개 short(2byte) x 2ch



88bytes, 44개 short(2bytes) x 2ch



※음악 출처 : <https://youtu.be/ciLNMesqPh0>

※참고 : <https://www.audacityteam.org>

※참고 : https://ko.wikipedia.org/wiki/2의_보수

메모리에 저장되는 데이터 – 음악 파일

❖ 메모리 == 파일 (0.0847s 구간, 0.001초만 WAV 파일로 저장)

Positions	Sample Value	Description
1-4	"RIFF"	Marks the file as a riff file. Characters are each 1 byte long.
5-8	File size (integer)	Size of the overall file - 8 bytes, in bytes (32-bit integer). Typically, you'd fill this in after creation.
9-12	"WAVE"	File Type Header. For our purposes, it always equals "WAVE".
13-16	"fmt "	Format chunk marker. Includes trailing null
17-20	16	Length of format data as listed above
21-22	1	Type of format (1 is PCM) - 2 byte integer
23-24	2	Number of Channels - 2 byte integer
25-28	44100	Sample Rate - 32 byte integer. Common values are 44100 (CD), 48000 (DAT). Sample Rate = Number of Samples per second, or Hertz.
29-32	176400	(Sample Rate * BitsPerSample * Channels) / 8.
33-34	4	(BitsPerSample * Channels) / 8.1 - 8 bit mono2 - 8 bit stereo/16 bit mono4 - 16 bit stereo
35-36	16	Bits per sample
37-40	"data"	"data" chunk header. Marks the beginning of the data section.
41-44	File size (data)	Size of the data section.

WAV File format

※ 참고 :

<https://docs.fileformat.com/audio/wav/>

Little endian : 0x000000B0 = 176byte, 447 short(2byte) x 2ch

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDE	F										
00:	52	49	46	46	04	00	00	00	57	41	56	45	66	6D	74	28	R	I	F	F	...	W	A	V	E	f	m	t
10:	10	00	00	00	01	00	02	00	44	AC	00	00	10	01	02	00	
20:	04	00	10	00	64	61	74	00	30	00	00	00	FD	06	A5	07	
30:	F3	09	A9	0A	DD	0C	CA	0D	03	0F	1E	10	18	10	51	11	
40:	23	10	3E	11	35	0F	37	10	67	0D	94	0E	0F	0B	91	0C	#	.	>	.	5	.	7	.	g	
50:	1B	08	F3	09	D9	04	DB	06	B0	01	C9	03	16	FF	1D	01	
60:	F6	FC	D6	FE	08	FB	EA	FC	DC	F8	F6	FA	32	F6	68	F8	
70:	15	F3	24	F5	D4	EF	8C	F1	A7	EC	06	EE	94	E9	C5	EA	
80:	CD	E6	14	E8	A3	E4	FC	E5	15	E3	68	E4	43	E2	51	E3	
90:	24	E2	BD	E2	63	E2	85	E2	70	E2	3A	E2	30	E2	73	E1	\$	
A0:	98	E1	3C	E0	BC	E0	D4	DE	AA	DF	82	DD	BB	DE	6C	DC	
B0:	3E	DE	96	DB	48	DE	47	DB	BE	DE	99	DB	DA	DF	7B	DC	>	
C0:	21	E2	35	DE	53	E5	F9	E0	5B	E8	E3	E3	60	EA	FB	E5	!	
D0:	A3	EB	19	E7	9A	EC	C5	E7	6C	ED	64	E8	

Little endian : 0x06fd
Ch : 0

Little endian : 0x07a5
Ch : 1

MSB==1 → 음수

※참고 : <https://hexed.it>

메모리에 저장되는 데이터 - 영상

❖ 변수형 : unsigned char - 컬러

※참고 : Hello-Vision (경희대 영상처리연구소)



Ch : R

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	
0	53	20	33	28	32	36	41	44	48	50	107	125	82	22	53	55	71	91	20	68	62	40	30	32	25	38
1	41	31	30	23	27	36	31	62	92	74	121	13	65	89	44	73	116	90	101	94	79	34	44	35	47	0
2	23	27	26	21	16	18	13	21	42	0	20	58	44	7	9	0	62	93	74	0	14	21	24	20	18	54
3	32	25	21	24	20	23	29	44	55	45	5	25	46	45	52	66	65	54	56	50	54	49	44	45	61	61
4	32	30	33	53	65	74	62	75	87	85	84	76	76	75	67	58	57	54	45	46	58	45	48	0	0	0
5	43	38	46	35	45	60	25	25	25	61	86	73	34	54	46	42	44	29	21	23	22	20	25	33	62	0
6	36	27	32	25	14	18	29	21	30	34	19	16	17	23	20	22	22	43	55	56	56	71	43	55	36	10
7	31	26	30	27	19	20	26	25	27	30	36	39	38	35	37	37	37	54	64	74	68	97	76	75	75	0
8	58	48	45	42	49	32	31	34	29	52	75	66	65	66	76	62	61	59	65	52	44	34	38	47	41	61
9	61	56	53	44	52	35	33	40	39	54	74	54	39	37	38	35	28	30	42	42	41	43	47	16	74	12
10	42	37	33	26	17	22	19	24	31	35	0	0	21	16	25	28	32	26	35	40	37	49	74	16	77	21
11	35	29	32	29	28	34	29	41	38	29	81	82	25	33	33	28	28	27	40	45	58	66	64	63	80	105
12	60	57	55	53	52	54	53	44	48	56	119	135	44	39	51	58	85	83	80	65	64	67	66	61	92	116
13	58	51	48	47	40	37	36	35	36	46	58	38	60	67	47	50	55	51	58	72	76	78	83	85	85	0
14	54	43	35	37	31	35	32	28	28	62	41	1	22	24	25	18	20	16	22	34	62	64	59	56	64	0
15	56	45	45	42	42	46	42	38	44	61	60	27	26	28	24	27	29	20	25	36	38	53	39	25	53	59
16	63	56	57	53	50	48	51	48	45	41	54	46	42	44	43	40	46	45	41	43	45	49	54	54	51	16
17	18	22	28	26	26	29	33	37	30	60	77	47	52	47	44	40	47	37	37	36	39	40	42	60	143	0
18	49	52	44	33	28	25	38	29	0	70	109	1	20	18	21	20	19	23	24	25	37	55	51	58	43	141
19	47	55	56	43	42	40	43	42	36	46	56	47	37	33	36	35	30	38	53	58	46	57	52	62	126	0
20	37	40	38	33	23	23	24	33	43	37	32	53	59	46	43	47	45	51	54	53	60	68	60	42	89	79
21	22	46	41	32	31	28	25	21	20	29	29	34	46	62	66	55	53	58	67	52	63	58	42	58	51	0
22	44	36	43	41	36	33	37	27	22	19	24	25	31	47	49	42	36	34	37	29	28	22	34	59	78	0
23	77	50	40	42	42	39	38	51	38	25	28	23	30	33	30	32	32	29	33	42	51	64	42	77	77	0
24	66	40	49	29	28	30	39	43	49	56	31	32	35	34	32	30	32	30	38	41	61	63	75	74	59	53
25	45	28	35	34	34	34	33	34	33	27	36	26	17	23	24	18	57	40	36	36	73	106	111	80	103	101

Ch : G

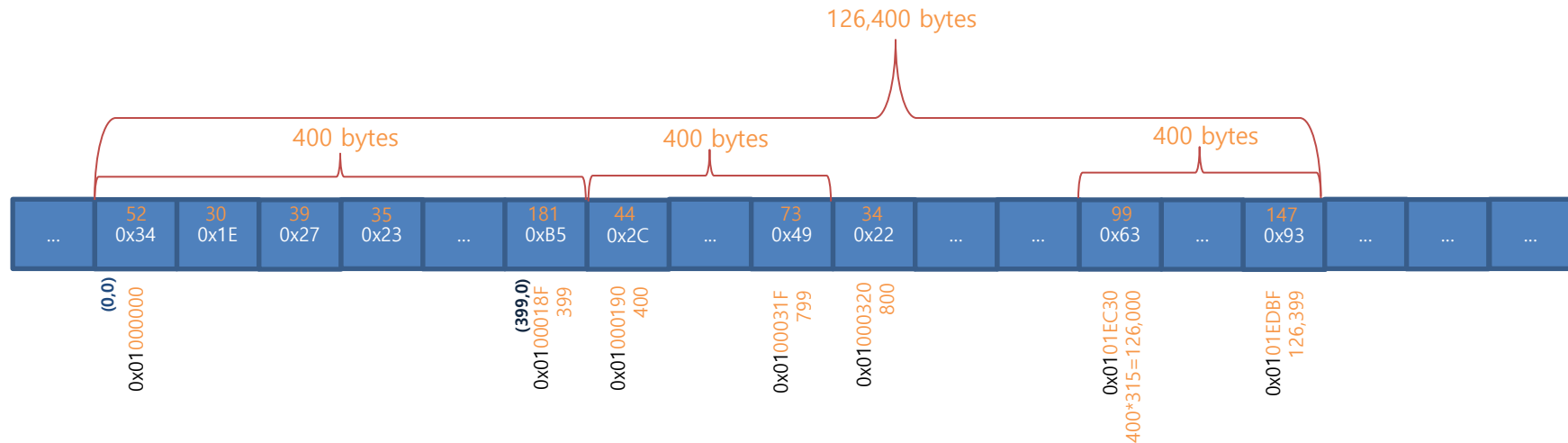
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	
0	50	29	38	34	37	38	45	47	52	106	106	124	83	34	61	60	81	98	33	75	71	44	39	40	32	46
1	41	37	36	28	29	31	39	33	66	92	73	101	11	71	89	52	75	122	102	105	99	82	49	52	44	54
2	31	31	30	26	14	17	16	25	41	0	28	63	47	19	10	0	74	106	83	7	23	29	22	24	61	
3	33	28	23	27	24	34	22	29	62	50	51	26	38	59	66	87	91	76	73	66	63	56	59	71	71	
4	35	31	31	26	57	66	78	64	81	95	98	97	91	92	86	73	72	66	58	60	69	59	64	60	0	
5	43	38	46	37	46	61	78	34	30	38	75	82	45	65	58	51	55	36	29	28	30	26	36	45	75	
6	37	34	40	31	20	26	37	56	47	41	20	20	22	30	30	27	56	72	73	71	83	56	69	56	109	
7	33	29	34	33	22	32	28	27	31	38	40	49	48	46	46	47	48	71	84	93	84	71	68	97	76	
8	67	57	54	48	56	29	37	42	37	64	88	81	82	83	73	72	74	70	67	59	49	51	59	53	75	
9	75	71	66	51	60	44	41	49	50	60	84	67	52	50	49	43	35	47	51	51	54	57	58	33	83	
10	51	48	40	33	25	26	21	26	36	44	0	0	21	19	29	33	40	33	46	48	47	62	86	36	88	
11	38	36	39	34	33	42	49	54	48	43	86	87	34	41	43	37	37	50	52	57	71	79	73	73	91	
12	76	74	70	69	72	72	57	64	74	80	34	53	51	61	71	95	95	94	81	80	84	83	81	105	105	
13	65	60	58	61	53	49	47	46	45	42	50	60	43	45	77	77	58	58	63	61	77	91	96	97	99	
14	58	49	39	43	35	41	37	33	30	68	55	17	25	29	31	27	30	25	32	43	78	75	75	63	78	
15	62	57	59	56	59	60	49	59	77	70	28	32	34	29	32	34	41	45	43	46	41	52	48	67	79	
16	68	72	77	72	71	67	67	63	61	55	65	62	59	59	57	66	63	54	58	61	58	63	70	61	62	
17	17	25	32	32	32	35	41	47	41	72	88	61	63	69	65	60	53	46	47	46	51	62	56	70	151	
18	57	63	55	59	31	28	34	0	73	14	21	20	32	22	19	17	27	30	30	44	71	72	73	65	146	
19	56	67	65	53	52	51	54	53	47	55	66	48	57	48	46	49	48	50	53	49	67	71	65	74	127	
20	44	48	46	41	29	29	39	34	47	46	74	62	62	67	65	70	70	68	80	79	57	56	94	92	92	
21	27	56	50	49	41	33	30	25	23	33	35	46	61	70	85	74	72	77	64	79	81	66	51	70	80	
22	67	44	56	69	59	41	45	33	25	29	28	38	59	61	51	42	40	43	35	29	26	39	41	73	81	
23	90	62	49	52	53	53	46	44	62	47	41	33	35	31	39	41	39	43	42	40	42	52	61	74	63	
24	75	51	49	35	37	45	50	62	70	38	40	44	41	41	40	39	40	46	46	66	70	81	85	77	67	
25	49	18	40	42	42	38	40	39	32	43	35	27	30	33	34	28	64	40	47	70	132	114	84	112	116	

Ch : B

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	
0	57	62	60	56	54	52	57	68	72	102	96	114	89	66	86	81	99	106	74	100	92	70	81	70	88	
1	66	80	78	61	61	54	61	55	80	74	106	107	91	99	75	87	105	119	132	111	104	97	89	94	99	
2	54	60	44	44	30	32	29	46	34	56	83	68	58	44	36	35	30	17	82	70	69	64	51	55	86	
3	62	69	40	52	47	53	41	50	75	72	64	61	72	89	84	93	124	129	110	118	109	109	95	93	102	113
4	62	61	53	63	71	69	88	75	91	103	117	118	118	118	123	127	141	114	107	111	106	109	124	109	61	
5	57	59	66	62	54	74	80	67	64	81	105	114	75	77	100	103	97	93	80	76	72	76	89	100	103	122
6	53	58	78	64	73	84	61	87	84	47	59	62	68	77	72	75	106	117	119	133	103	116	124	133	151	
7	50	45	61	73	55	74	69	56	60	64	73	79	80	72	71	74	86	115	125	134	126	134	126	134	151	
8	57	59	66	62	54	74	80	67	64	81	105	114	75	77	100	103	97	93	80	76	72	76	89	100	103	122
9	53	58	78	64	73	84	61	87	84	47	59	62	68	77	72	75	106	117	119	133	103	116	124	133	151	
10	50	45	61	73	55	74	69	56	60	64	73	79	80	72	71	74	86	115	125	134	126	134	126	134	151	
11	57	59	66	62	54	74	80	67	64	81	105	114	75	77	100	103	97	93	80	76	72	76	89	100	103	122
12	53	58	78	64	73	84	61	87	84	47	59	62	68	77	72	75	106	117	119	133	103	116	124	133	151	
13	50	45	61	73	55	74	69	56	60	64	73	79	80	72	71	74	86	115	125	134	126	134	126	134	151	
14	57	59	66	62	54	74	80	67	64	81	105	114	75	77	100	103	97	93	80	76	72	76	89	100	103	122
15	53	58	78	64	73	84	61	87	84	47	59	62	68	77	72	75	106	117	119	133	103	116	124	133	151	
16	50	45	61	73	55	74	69	56	60	64	73	79	80	72	71	74	86	115	125	134	126	134	126	134	151	
17	57	59	66	62	54	74	80	67	64	81	105	114	75	77	100	103	97	93	80	76	72	76	89	100	103	122
18	53	58	78	64	73	84	61	87	84	47	59	62	68	77	72	75	106	117	119	133	103	116	124	133	151	
19	50	45	61	73	55	74	69	56	60	64	73	79	80	72	71	74	86	115	125	134	126	134	126	134	151	
20	57	59	66	62	54	74	80	67	64	81	105	114	75	77	100	103	97	93	80	76	72	76	89	100	103	122
21	53	58	78	64	73	84	61	87	84	47	59	62	68	77	72	75	106	117	119	133	103	116	124	133	151	
22	50	45	61	73	55	74	69	56	60	64	73	79	80	72	71	74	86	115	125	134	126	134	126	134	151	
23	57	59	66	62	54	74	80	67	64	81	105	114	75	77	100	103	97	93	80	76	72	76	89	100	103	122
24	53	58	78	64	73	84	61	87	84	47	59	62	68	77	72	75	106	117	119	133	103	116	124	133	151	
25	50	45	61	73	55	74	69	56	60	64	73	79	80	72	71	74	86	115	125	134	126	134	126	134	151	

메모리에 저장되는 데이터 - 영상

❖ 메모리에 저장된 영상



❖ Windows BITMAP 형식의 영상



파일 끝 주소 : 1,078(파일헤더크기) + 126,399(영상데이터 끝 주소)
= 127,477
= 0x0001F1F5

※참고 : <https://docs.fileformat.com/image/bmp/>

※참고 : <https://docs.fileformat.com/image/bmp/>

00000000 42 4D F6 F1 01 00 00 00 00 00 36 04 00 00 28 00 BM±.....6..(.

00000010 00 00 90 01 00 00 3C 01 00 00 01 00 08 00 00 00 ..E...<.....

00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00<.....

00000030 00 00 00 00 00 00 00 00 00 00 01 01 01 00 02 02<.....

00000040 02 00 03 03 03 00 04 04 04 00 05 05 05 00 06 06<.....

00000050 06 00 07 07 07 00 08 08 08 00 09 09 09 00 0A 0A<.....

00000060 0A 00 0B 0B 0B 00 0C 0C 0C 00 0D 0D 0D 00 0E 0E<.....

00000070 0E 00 0F 0F 0F 00 10 10 10 00 11 11 11 00 12 12<.....

00000080 12 00 13 13 13 00 14 14 14 00 15 15 15 00 16 16<.....

00000090 16 00 17 17 17 00 18 18 18 00 19 19 19 00 1A 1A<.....

000000A0 1A 00 1B 1B 1B 00 1C 1C 1C 00 1D 1D 1D 00 1E 1E<.....

000000B0 1E 00 1F 1F 1F 00 20 20 20 00 21 21 21 00 22 22<.....

000000C0 22 00 23 23 23 00 24 24 24 00 25 25 25 00 26 26<.....

000000D0 26 00 27 27 27 00 28 28 28 00 29 29 29 00 2A 2A<.....

000000E0 2A 00 2B 2B 2B 00 2C 2C 2C 00 2D 2D 2D 00 2E 2E<.....

000000F0 2E 00 2F 2F 2F 00 30 30 30 00 31 31 31 00 32 32<.....

00000100 32 00 33 33 33 00 34 34 34 00 35 35 35 00 36 36<.....

00000110 36 00 37 37 37 00 38 38 38 00 39 39 39 00 3A 3A<.....

00000120 3A 00 3B 3B 3B 00 3C 3C 3C 00 3D 3D 3D 00 3E 3E<.....

00000130 3E 00 3F 3F 3F 00 40 40 40 00 41 41 41 00 42 42<.....

00000140 42 00 43 43 43 00 44 44 44 00 45 45 45 00 46 46<.....

00000150 46 00 47 47 47 00 48 48 48 00 49 49 49 00 4A 4A<.....

00000160 4A 00 4B 4B 4B 00 4C 4C 4C 00 4D 4D 4D 00 4E 4E<.....

00000170 4E 00 4F 4F 4F 00 50 50 50 00 51 51 51 00 52 52<.....

00000180 52 00 53 53 53 00 54 54 54 00 55 55 55 00 56 56<.....

00000190 56 00 57 57 57 00 58 58 58 00 59 59 59 00 5A 5A<.....

000001A0 5A 00 5B 5B 5B 00 5C 5C 5C 00 5D 5D 5D 00 5E 5E<.....

000001B0 5E 00 5F 5F 5F 00 60 60 60 00 61 61 61 00 62 62<.....

000001C0 62 00 63 63 63 00 64 64 64 00 65 65 65 00 66 66<.....

000001D0 66 00 67 67 67 00 68 68 68 00 69 69 69 00 6A 6A<.....

000001E0 6A 00 6B 6B 6B 00 6C 6C 6C 00 6D 6D 6D 00 6E 6E<.....

000001F0 6E 00 6F 6F 6F 00 70 70 70 00 71 71 71 00 72 72<.....

00000200 72 00 73 73 73 00 74 74 74 00 75 75 75 00 76 76<.....

00000210 76 00 77 77 77 00 78 78 78 00 79 79 79 00 7A 7A<.....

00000220 7A 00 7B 7B 7B 00 7C 7C 7C 00 7D 7D 7D 00 7E 7E<.....

00000230 7E 00 7F 7F 7F 00 80 80 80 00 81 81 81 00 82 82<.....

00000240 82 00 83 83 83 00 84 84 84 00 85 85 85 00 86 86<.....

00000250 86 00 87 87 87 00 88 88 88 00 89 89 89 00 8A 8A<.....

00000260 8A 00 8B 8B 8B 00 8C 8C 8C 00 8D 8D 8D 00 8E 8E<.....

00000270 8E 00 8F 8F 8F 00 90 90 90 00 91 91 91 00 92 92<.....

00000280 92 00 93 93 93 00 94 94 94 00 95 95 95 00 96 96<.....

00000290 96 00 97 97 97 00 98 98 98 00 99 99 99 00 9A 9A<.....

000002A0 9A 00 9B 9B 9B 00 9C 9C 9C 00 9D 9D 9D 00 9E 9E<.....

000002B0 9E 00 9F 9F 9F 00 A0 A0 A0 00 A1 A1 A1 00 A2 A2<.....

000002C0 A2 00 A3 A3 A3 00 A4 A4 A4 00 A5 A5 A5 00 A6 A6<.....

000002D0 A6 00 A7 A7 A7 00 A8 A8 A8 00 A9 A9 A9 00 AA AA<.....

000002E0 AA 00 AB AB AB 00 AC AC AC 00 AD AD AD 00 AE AE<.....

000002F0 AE 00 AF AF AF 00 B0 B0 B0 00 B1 B1 B1 00 B2 B2<.....

00000300 B2 00 B3 B3 B3 00 B4 B4 B4 00 B5 B5 B5 00 B6 B6<.....

00000310 B6 00 B7 B7 B7 00 B8 B8 B8 00 B9 B9 B9 00 BA BA<.....

00000320 BA 00 BB BB BB 00 BC BC BC 00 BD BD BD 00 BE BE<.....

00000330 BE 00 BF BF BF 00 C0 C0 C0 00 C1 C1 C1 00 C2 C2<.....

00000340 C2 00 C3 C3 C3 00 C4 C4 C4 00 C5 C5 C5 00 C6 C6<.....

00000350 C6 00 C7 C7 C7 00 C8 C8 C8 00 C9 C9 C9 00 CA CA<.....

00000360 CA 00 CB CB CB 00 CC CC CC 00 CD CD CD 00 CE CE<.....

00000370 CE 00 CF CF CF 00 D0 D0 D0 00 D1 D1 D1 00 D2 D2<.....

00000380 D2 00 D3 D3 D3 00 D4 D4 D4 00 D5 D5 D5 00 D6 D6<.....

00000390 D6 00 D7 D7 D7 00 D8 D8 D8 00 D9 D9 D9 00 DA DA<.....

000003A0 DA 00 DB DB DB 00 DC DC DC 00 DD DD DD 00 DE DE<.....

000003B0 DE 00 DF DF DF 00 E0 E0 E0 00 E1 E1 E1 00 E2 E2<.....

000003C0 E2 00 E3 E3 E3 00 E4 E4 E4 00 E5 E5 E5 00 E6 E6<.....

000003D0 E6 00 E7 E7 E7 00 E8 E8 E8 00 E9 E9 E9 00 EA EA<.....

000003E0 EA 00 EB EB EB 00 EC EC EC 00 ED ED ED 00 EE EE<.....

000003F0 EE 00 EF EF EF 00 F0 F0 F0 00 F1 F1 F1 00 F2 F2<.....

00000400 F2 00 F3 F3 F3 00 F4 F4 F4 00 F5 F5 F5 00 F6 F6<.....

0

(399,315)
파일 끝 주소 = 0x0001F1F5
Value = 0xB5 = 181

(0,0) = 1,078 = 0x00000436
Value = 0x63 = 99

	0.	1.	2.	3.	4.	5.	6.	
00000430	FE	00	FF	FF	FF	63	5E	48 32 2E 3C 54 6C 76 64 . . . 2.H2.<Tlvd
00000440	60	72	87	7E	2D	26	7F	33 3C 71 8B 85 7E 93 99 9B 'rç--&/3<qia=00c
00000450	81	70	82	8C	82	85	96	4F 80 09 6B 7E 33 81 3E üpetéâû°C.k->ôû
00000460	A0	75	47	60	66	6A	53	5B 64 55 4B 48 52 58 53 70 äUG fj\$[dUKKR[Sp
00000470	90	92	92	84	78	63	62	64 54 44 3D 2F 0D 4C 6E 33 É&äxcdbTD=/.Ln3
00000480	22	21	29	1C	25	23	1E	1B 1E 1A 10 11 24 21 14 26 "!)%#...\$!&.
00000490	31	1F	15	29	49	3B	22	46 28 18 10 20 27 16 1B 32 1..)";"FE("'.2
000004A0	43	27	1D	28	17	19	26	31 2A 20 21 44 2D 1C 2C 16 C.().&1* !D=.
000004B0	1B	13	5B	4B	06	24	1E	1D 15 30 27 11 22 2B 1D 1C ..[K.\$...0'"+
000004C0	3D	4A	31	35	04	28	67	5F 03 49 22 24 20 17 19 4C =J15.(g..I'\$..L
000004D0	39	19	1A	16	21	2D	0F	24 1D 19 16 19 18 12 17 26 9...!-.\$.....&
000004E0	41	37	36	3A	19	2C	14	24 28 29 27 26 04 76 6C 1D A76:..,\$(')&.v!
000004F0	38	2B	2C	41	40	52	49	40 3A 2A 3E 45 44 50 43 48 8+,@&R!@:;>EDPCH
00000500	51	45	46	4A	30	33	3B	31 39 3D 41 3E 32 36 40 3D QEFJ=3;19=A>2G6=
00000510	3D	3E	3D	3E	44	44	3F	39 3D 43 3A 34 3D 2E 25 25 =>>DD?9=C:4=.%%

(0,399) : 127,478-400 = 127,078
= 0x0001F066, Value = 0x34 = 52

	0.	1.	2.	3.	4.	5.	6.	
0001F060	2A	3B	A0	B4	9C	34	1E	27 23 25 27 2D 30 35 69 *;â-É4.'#%-05i
0001F070	69	7B	53	22	3D	30	50	61 22 4C 47 2E 29 22 30 i{S'=Pa"LG.))"0
0001F080	20	5B	68	67	42	47	49	54 62 62 78 88 74 88 7F [jhgBGITbxbxê&e
0001F090	63	68	71	85	74	74	72	83 6C 1C 18 3C 54 26 2F 1 hhqâztrâl..sT&/1
0001F0A0	27	1E	29	36	45	4D	44	2B 4A 65 3C 1F 05 39 50 46 '.)6EMD+Je<..9PF
0001F0B0	3A	39	27	30	61	72	75	6C 83 80 85 73 38 40 50 4A 9'0arula&âs8&PJ
0001F0C0	47	48	3E	4C	49	34	36	39 42 47 48 47 39 3A 39 31 GH>LI469BGHG9:91
0001F0D0	25	21	37	6B	48	32	17	30 37 35 19 85 82 5D 45 4A %!7kH2.075.âé]EJ
0001F0E0	64	42	3B	49	67	72	78	5B 3C 30 41 5E 68 73 89 68 dB;Igrx[<0A'hsh&h
0001F0F0	6A	76	7B	72	5B	57	55	45 58 3C 26 39 1F 64 8E 6D jv{[WUKX<&9.d&M
0001F100	4A	4F	3C	3E	5C	72	79	74 62 5D 5A 4F 49 4B 4E 43 J0<[>rytbjZOIKNC
0001F110	42	3E	4A	43	41	3F	5A	81 76 66 73 69 71 77 6C 66 B>JCA?ZüvfsiqwlF
0001F120	62	63	62	6F	75	7F	71	58 54 5E 5D 73 6B 66 45 5B bcbou&qT&A]skf[
0001F130	73	7A	72	79	73	68	66	79 61 4B 50 52 58 6B 71 6F szryshfy&KPRXmkp
0001F140	6A	6F	64	61	7E	62	5F	5D 55 51 57 4D 40 50 3D 33 joda-b.]UQWMMp=3
0001F150	34	34	32	39	43	51	5D	61 64 62 57 65 6C 6E 62 55 4429CQJadbw&Inbu
0001F160	5C	3E	29	5E	6C	51	56	65 56 71 62 65 6C 6E 76 73 \>)^\QV&f&bellvs
0001F170	5E	67	70	6D	63	54	53	57 5B 5A 52 5C 52 5B 76 6F ^gpmcSTW[Z&R'Y[fo
0001F180	69	57	5F	66	66	58	43	3E 46 51 43 3F 42 4A 4D 43 iW_ffXC>FQC?BJMC
0001F190	5D	5E	66	80	8E	83	7	

변수 자료형에 따른 값의 범위

자료형	크기(byte)	진법	범위					비고
			최소값	중간값			최대값	
int	4	DEC	-2,147,483,648	-1	0	1	2,147,483,647	
		HEX	0x80000000	0xFFFFFFFF	0x00000000	0x00000001	0x7FFFFFFF	
unsigned int	4	DEC	0	2,147,483,646	2147483647	2,147,483,648	4,294,967,295	
		HEX	0x00000000	0x7FFFFFFE	0x7FFFFFFF	0x80000000	0xFFFFFFFF	
char	1	DEC	-128	-1	0	1	127	ANSI 문자
		HEX	0x80	0xFF	0x00	0x01	0x7F	
unsigned char	1	DEC	0	126	127	128	255	
		HEX	0x00	0x7E	0x7F	0x80	0xFF	
short	2	DEC	-32,768	-1	0	1	32,767	
		HEX	0x8000	0xFFFF	0x0000	0x0001	0x7FFF	
unsigned short	2	DEC	0	32766	32767	32,768	65,535	
		HEX	0x0000	0x7FFE	0x7FFF	0x8000	0xFFFF	

※ OS 등의 시스템마다 다를 수 있음

❖ 연산 예시

나누기와 곱하기의 연산을 수행할 때는 많은 주의가 필요

```
01 int    nC;
02 double dC;
03
04 nC = 5 / 2 ; // 변수 nC의 저장 값은? 2
05 dC = 5.0 / 2 ; // 변수 dC의 저장 값은? 2.5
06 nC = 10000000 * 10000000 ; // 변수 nC의 저장 값은?
07 nC = -1000000000 * 1000000000 ; // 변수 nC의 저장 값은?
```

소스를 보면 동일하게 나누기를 수행했는데도 출력 값이 다른
기본으로 피연산자 중에서 데이터 타입이 큰 쪽을 따라 결과를 출력하도록 설계

- 정수 나누기 정수의 결과는 정수다.
- 실수 나누기 정수의 결과는 실수다.

❖ 나누기와 곱하기 연산을 수행할 때 명심할 사항

- 정수와 정수의 연산결과는 항상 정수다.
- 정수와 실수의 연산결과는 항상 실수다.
- 산술연산을 할 때는 연산결과, 버림 현상이 발생하지 않는지 확인한다.
- float 데이터 타입을 사용해 나눗셈을 수행할 때 소수점 이하의 오차율이 커지는 문제가 발생하지 않는지 확인한다.
- 곱하기 연산을 할 때 **오버플로**나 **언더플로**가 발생하지 않는지 확인한다.

2. C++에서 변수 선언과 메모리 할당

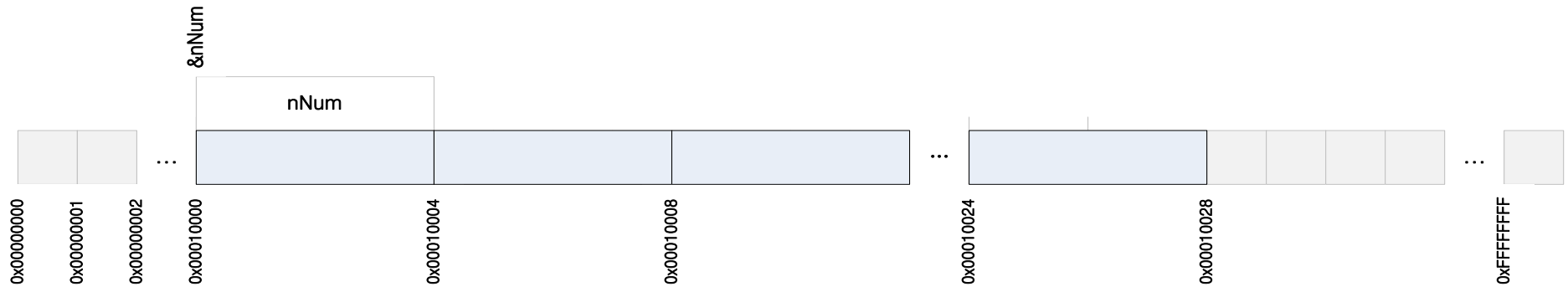
(32bit 시스템 가정)

- 변수, 배열과 메모리
- 포인터 변수와 메모리
- 배열, 포인터와 메모리
- 배열과 포인터
- 동적 메모리 할당
- 포인터 연산
- 2차원 동적 할당

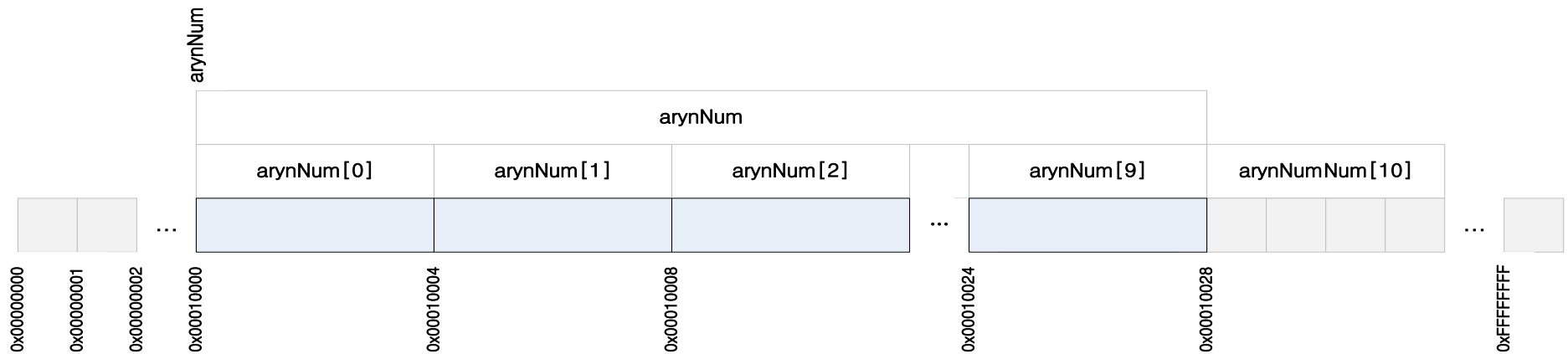


변수, 배열과 메모리

```
int nNum;
```

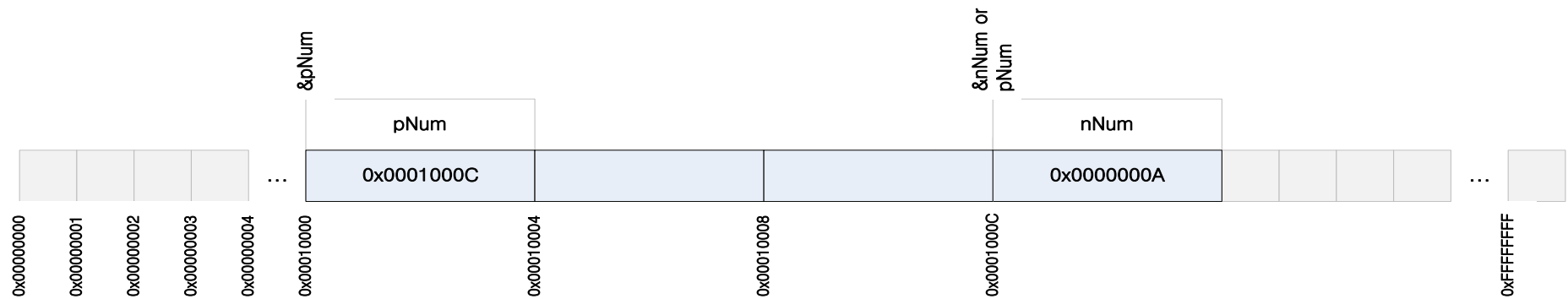


```
int arynNum[10];
```

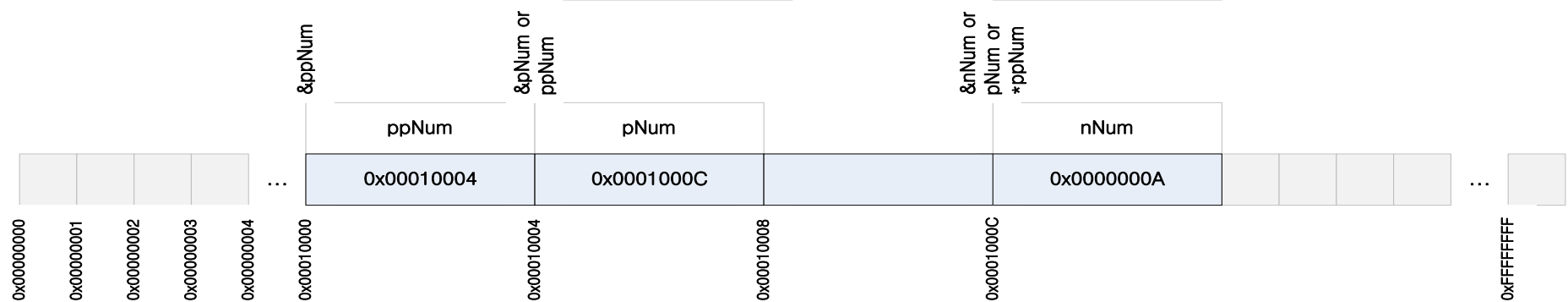


포인터 변수와 메모리

```
int nNum = 10;
int* pNum = &nNum; // or int *pNum = &nNum;
```

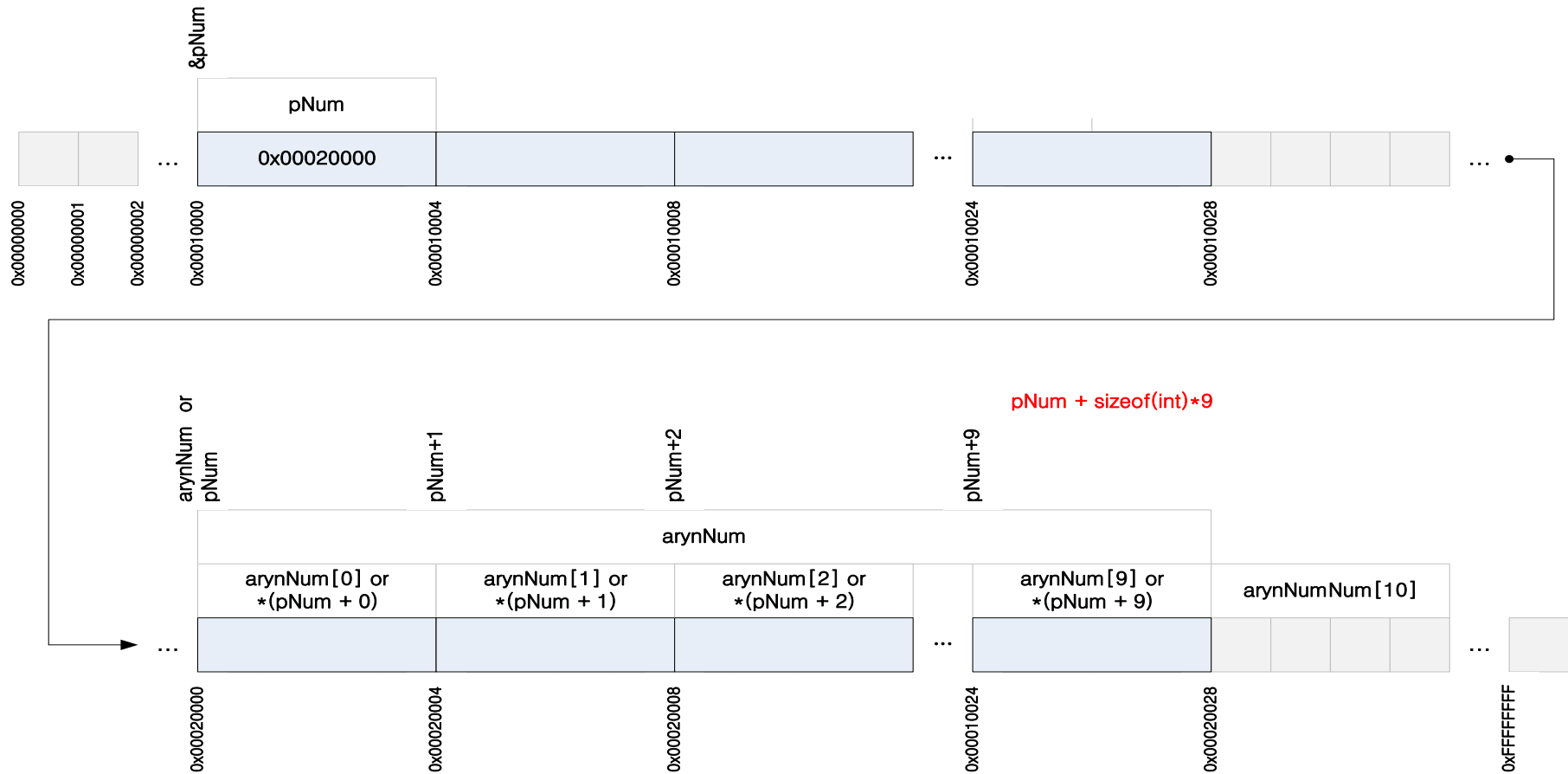


```
int nNum = 10;
int* pNum = &nNum;
int** ppNum = &pNum;
```



배열, 포인터와 메모리

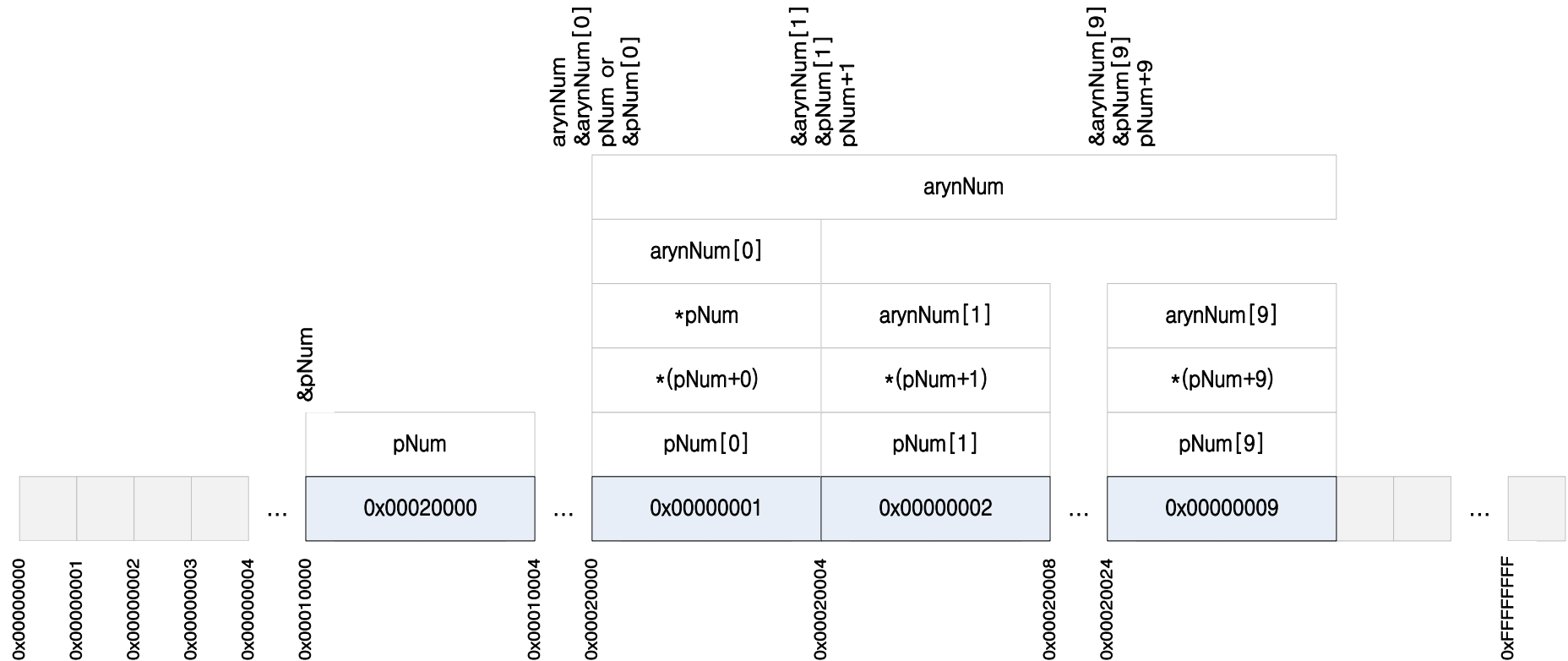
```
int arynNum[10];  
int* pNum = arynNum;
```



&arynNum ???

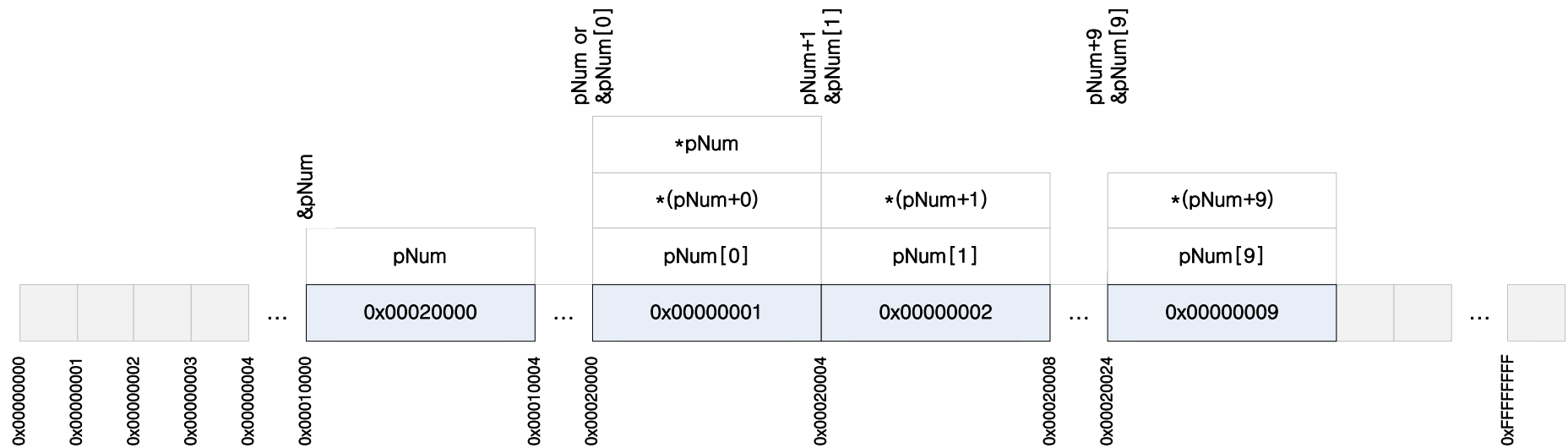
배열과 포인터

```
int arynNum[10];  
Int* pNum      = arynNum;  
*pNum          = 1;  
  arynNum[1]   = 2;  
*(pNum+9)      = 9;
```



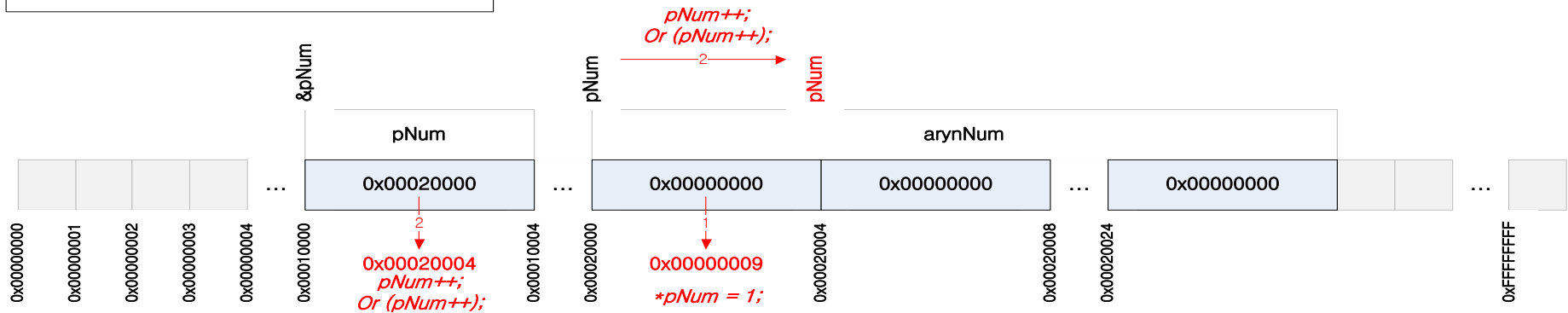
동적 메모리 할당

```
int* pNum = new int[10];  
*pNum = 1;  
pNum[1] = 2;  
*(pNum+9) = 9;  
delete[] pNum;
```

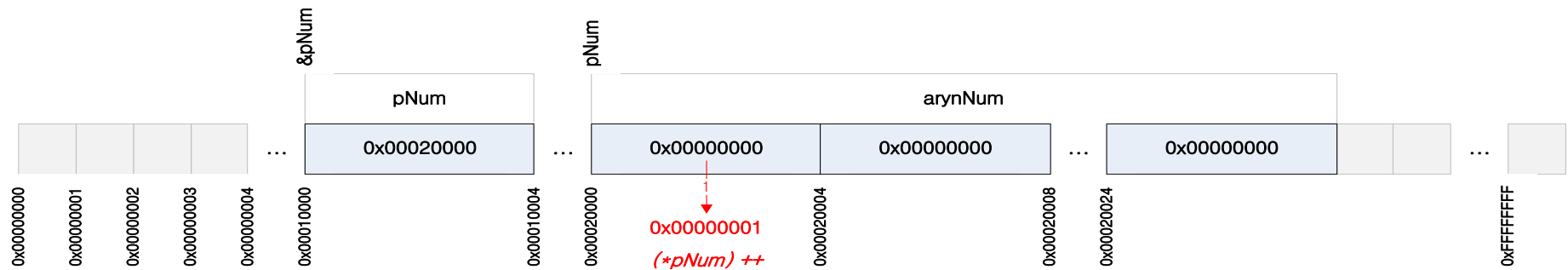


포인터 연산

```
int arynNum[10]={0};
int* pNum    = arynNum;
*pNum++      = 9; // or *(pNum++) = 9;
```



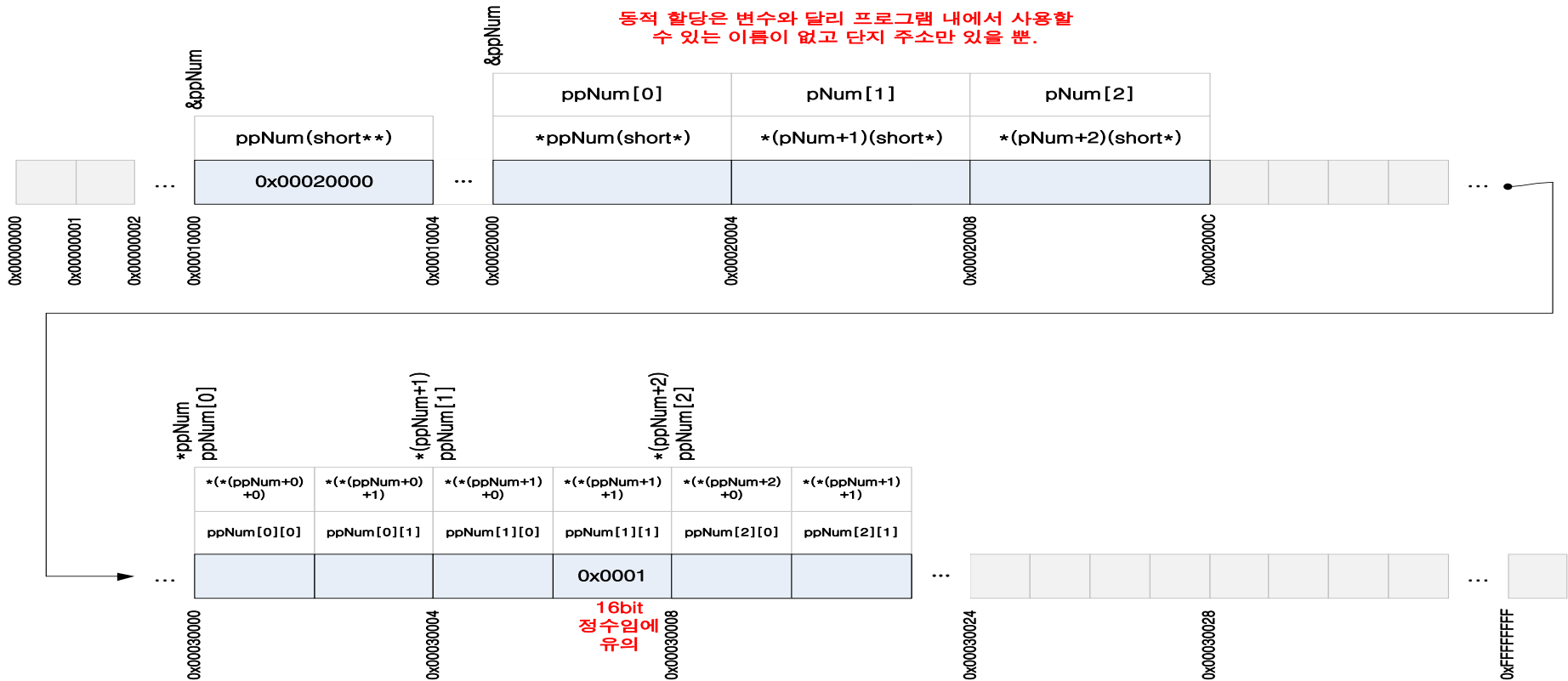
```
int arynNum[10]={0};
int* pNum    = arynNum;
(*pNum)++;
```



2차원 동적 할당

```
short** ppNum = new short*[3];
for( int i=0 ; i<3 ; i++ )
    *(ppNum+i) = new short[2]; // or ppNum[i] = new short[2];
*(*(ppNum+1)+1) = 1;
for( int i=0 ; i<3 ; i++ )
    delete *(ppNum+i); //or delete ppNum[i];
delete ppNum;
```

동적 할당은 변수와 달리 프로그램 내에서 사용할 수 있는 이름이 없고 단지 주소만 있을 뿐.



3. 데이터와 메모리,파일 연습

- 사인파 오디오 파일



메모리에 저장되는 데이터 – 사인파 오디오 파일

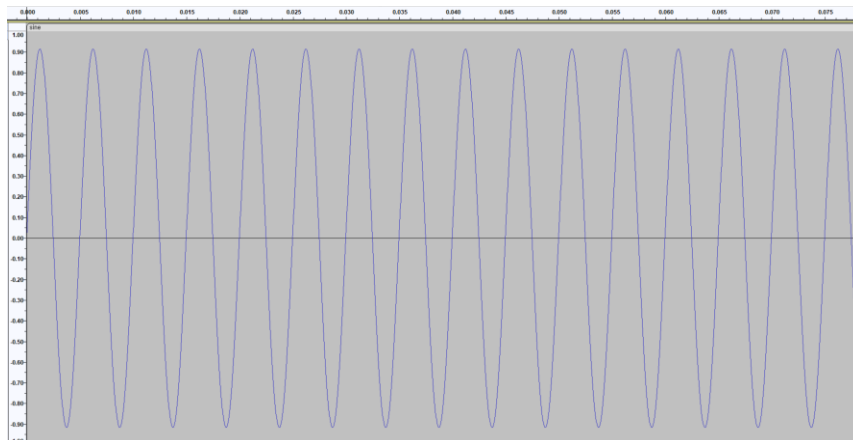
❖ Sine WAV Audio File



Positions	Sample Value	Description
1-4	"RIFF"	Marks the file as a riff file. Characters are each 1 byte long.
5-8	File size (integer)	Size of the overall file - 8 bytes, in bytes (32-bit integer). Typically, you'd fill this in after creation.
9-12	"WAVE"	File Type Header. For our purposes, it always equals "WAVE".
13-16	"fmt "	Format chunk marker. Includes trailing null
17-20	16	Length of format data as listed above
21-22	1	Type of format (1 is PCM) - 2 byte integer
23-24	2	Number of Channels - 2 byte integer
25-28	44100	Sample Rate - 32 byte integer. Common values are 44100 (CD), 48000 (DAT). Sample Rate = Number of Samples per second, or Hertz.
29-32	176400	(Sample Rate * BitsPerSample * Channels) / 8.
33-34	4	(BitsPerSample * Channels) / 8.
35-36	16	Bits per sample
37-40	"data"	"data" chunk header. Marks the beginning of the data section.
41-44	File size (data)	Size of the data section.

Sine.wav : <MedialInfo>

Audio
 Format : PCM
 Format settings : Little / Signed
 Codec ID : 1
 Duration : 10 s 0 ms
 Bit rate mode : Constant
 Bit rate : 705.6 kb/s
 Channel(s) : 1 channel
 Sampling rate : 44.1 kHz
 Bit depth : 16 bits
 Stream size : 861 KiB (50%)



$$\text{Data Size} = 44100/1s * 10s * 2\text{byte}(\text{short}) = 882,000 = 0x000D7550$$

```

00000000  52 49 46 46 7C 75 0D 00 57 41 56 45 66 6D 74 20  RIFF|u..WAVEfmt
00000010  10 00 00 00 01 00 01 00 44 AC 00 00 88 58 01 00  ....D%.èX..
00000020  02 00 10 00 64 C1 74 61 50 75 0D 00 40 00 AB 01  ....dataPu...µ
00000030  56 03 01 05 AC 06 57 08 01 0A AB 0B 54 0D FC 0E  V...%.W...%.T..
00000040  A3 10 4A 12 F0 13 94 15 38 17 DA 18 7B 1A 1B 1C  ú.J.=.ö.8.Γ{...
00000050  B9 1D 56 1F F1 20 8A 22 22 24 B7 25 4B 27 DD 28  ¶.V.± è""$%K'|(
00000060  6C 2A FA 2B 85 2D 0D 2F 93 30 17 32 98 33 17 35  ɿ*+·à-./ô0.2ý3.5
    
```

$$\text{File Size} = \text{Data Size} + 44\text{bytes}(\text{Header}) = 882,044 = 0x000D757C$$

```

000D7530  7B C3 EA C4 5D C6 D2 C7 4B C9 C6 CA 45 CC C5 CD  {[-]KfE|=-
000D7540  49 CE CF D0 E7 D2 E2 D3 6F D5 FF D6 90 D8 24 DA  I=llWToF rE+$r
000D7550  B9 DB 50 DD EA DE 84 E0 21 E2 BF E3 5F E5 FF E6  ¶PΩ!α!Γπ_σµ
000D7560  A2 E8 45 EA EA EB 8F ED 36 EF DD F0 86 F2 2E F4  óφΕΩΩδÀφ6ñ=áz.ɿ
000D7570  D8 F5 82 F7 2D F9 D7 FA 83 FC 2E FE  ¶|éz~·.†·âñ..█
    0. 1. 2. 3. 4. 5. 6. 7. 8. 9. A. B. C
    
```

$$\text{File Address} = 0x00000000 \sim 0x000D757B$$

```
//Cerate By Kiok Ahn, kiokahn@gazzi.ai
//Home : http://gazzi.ai

//https://docs.fileformat.com/audio/wav/
//https://videolan.videolan.me/vlc/vlc__codecs_8h.html
```

```
#pragma once
```

```
#define WAVE_FORMAT_UNKNOWN    0x0000 /* Microsoft Corporation */
#define WAVE_FORMAT_PCM        0x0001 /* Microsoft Corporation */
#define WAVE_FORMAT_ADPCM      0x0002 /* Microsoft Corporation */
#define WAVE_FORMAT_IEEE_FLOAT 0x0003 /* Microsoft Corporation */
#define WAVE_FORMAT_ALAW       0x0006 /* Microsoft Corporation */
#define WAVE_FORMAT_MULAW      0x0007 /* Microsoft Corporation */
#define WAVE_FORMAT_DTS        0x0008 /* Microsoft Corporation */
#define WAVE_FORMAT_WMAS       0x000a /* WMA 9 Speech */
#define WAVE_FORMAT_IMA_ADPCM   0x0011 /* Intel Corporation */
#define WAVE_FORMAT_YAMAHA_ADPCM 0x0020 /* Yamaha */
#define WAVE_FORMAT_TRUESPEECH 0x0022 /* TrueSpeech */
#define WAVE_FORMAT_GSM610     0x0031 /* Microsoft Corporation */
#define WAVE_FORMAT_MSNAUDIO    0x0032 /* Microsoft Corporation */
#define WAVE_FORMAT_AMR_NB_2    0x0038 /* AMR NB rogue */
#define WAVE_FORMAT_MSG723      0x0042 /* Microsoft G.723 [G723.1] */
#define WAVE_FORMAT_SHARP_G726  0x0045 /* ITU-T standard */
#define WAVE_FORMAT_MPEG        0x0050 /* Microsoft Corporation */
// ...
```

```
typedef struct
```

```
{
    char    Riff    [4];

    // Marks the file as a riff file. Characters are each 1 byte long.

    unsigned int    FileSize    ;

    // Size of the overall file - 8 bytes, in bytes (32-bit integer).
    // Typically, you'd fill this in after creation.

    char    FileType  [4];

    // File Type Header. For our purposes, it always equals "WAVE".

    char    ChunkMarker [4];

    // Format chunk marker. Includes trailing null

    unsigned int    FormatLength ;// Length of format data as listed above

    unsigned short PCMFormat    ;

    // Type of format (1 is PCM) - 2 byte integer, WAVE_FORMAT_xxx

    unsigned short Channels    ;

    // Number of Channels - 2 byte integer, Mono = 1, Stereo = 2, etc.

    unsigned int    SampleRate    ;

    // Sample Rate - 32 byte integer. Common values are 44100 (CD), 48000 (DAT).
    // Sample Rate = Number of Samples per second, or Hertz.

    unsigned int    AvgByteRate ;// SampleRate * Channels * BitsPerSample/8

    unsigned short BlockAlign    ;

    unsigned short BitPerSample    ;

    // Bits per sample, 8 bits = 8, 16 bits = 16, etc

    char    ChunkDATA [4];

    // "data" chunk header. Marks the beginning of the data section.

    unsigned int    DataSize    ;// Size of the data section.

} WAV_HEADER;
```

Wav_Sine.cpp

```
#include <iostream>
#include <math.h>
#include "WavHeader.h"

#define DURATION 10 //(second)
#define SAMPLE_RATE 44100 //(Hz)
#define SINE_FREQUENCY 100 //(Hz)
#define AMPLITUDE 30000 //range of short, -32,768~32,767
#define CHANNEL 1
#define BIT_RATE 16 //16bit-1ch
#define PI 3.141592

int main(void)
{
    FILE * f_sine;

    f_sine = fopen("./sine.wav", "wb");

    int datasize = DURATION * SAMPLE_RATE * CHANNEL * BIT_RATE / 8;

    WAV_HEADER header;

    memcpy(header.Riff, "RIFF", sizeof(header.Riff));

    header.FileSize = datasize + sizeof(WAV_HEADER);

    memcpy(header.FileType, "WAVE", sizeof(header.FileType));

    memcpy(header.ChunkMarker, "fmt ", sizeof(header.ChunkMarker));

    header.FormatLength = 0x10;
```

```
    header.PCMFormat = WAVE_FORMAT_PCM;

    header.Channels = CHANNEL;

    header.SampleRate = SAMPLE_RATE;

    header.AvgByteRate = SAMPLE_RATE * CHANNEL * BIT_RATE / 8;

    header.BlockAlign = CHANNEL * BIT_RATE / 8;

    header.BitPerSample = BIT_RATE;

    memcpy(header.ChunkDATA, "data", sizeof(header.ChunkDATA));

    header.DataSize = datasize;

    fwrite(&header, sizeof(WAV_HEADER), 1, f_sine);

    short* wav_sine = new short[datasize];

    memset(wav_sine, 0, sizeof(short)*datasize);

    short* itor = wav_sine;

    for( int i = 0; i < datasize; i++ )
    {
        *itor++ = (short)AMPLITUDE*sin(2 * PI * i * SINE_FREQUENCY
            / SAMPLE_RATE);
    }

    fwrite(wav_sine, 1, datasize, f_sine);

    fclose(f_sine);

    delete[] wav_sine;

    return 0;
}
```