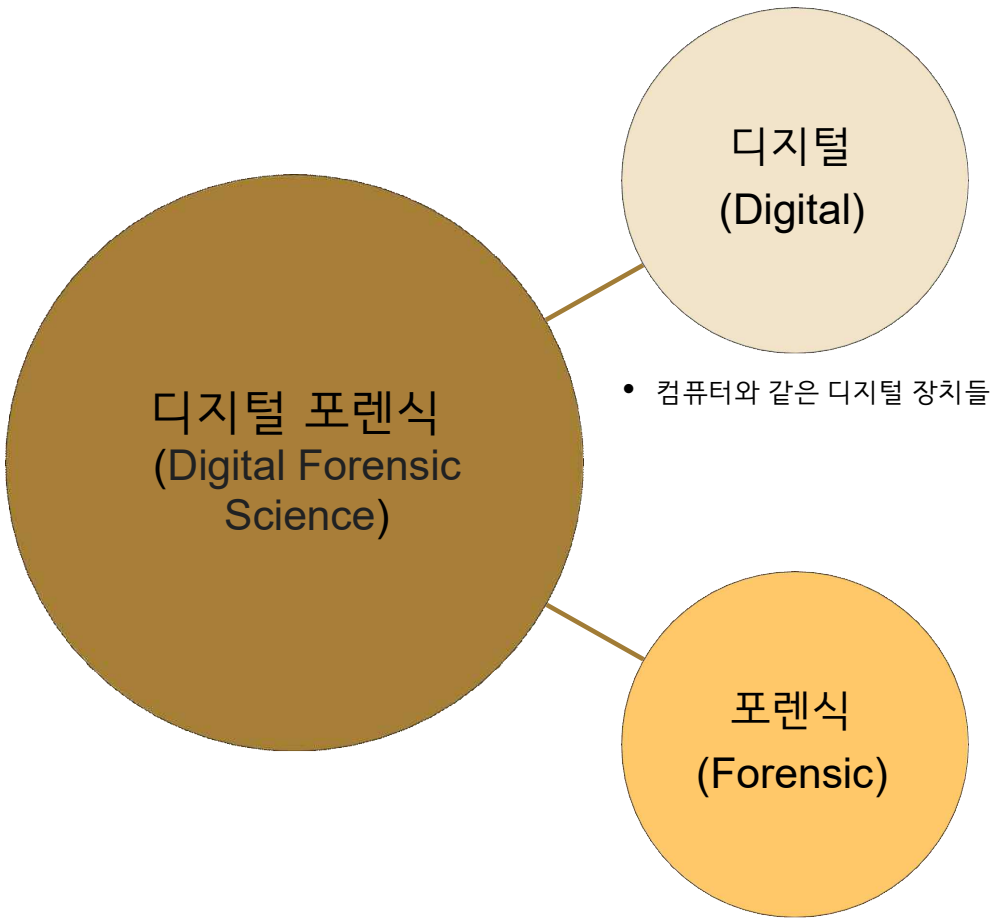
ZIP 파일 분석

20152003 옥정수

목차

1. 디지털 포렌식이란?



디지털 포렌식(Digital Forensic)은 PC나 핸드폰 등 디지털 기기에 남아있는 데이터를 토대로 범죄의 단서나 증거를 찾아내는 과학 수사 기법이다.

포렌식(forensic)은 ‘법의학의’, ‘법정의’와 같은 사전적인 의미를 지니고 있는데 법의학처럼 디지털 기기를 파헤쳐 법정에서 유효한 증거를 도출해 내는 것이 바로 **디지털 포렌식**이라고 할 수 있다.

2. 디지털 포렌식 종류

디스크 포렌식 : 흔히 접할 수 있는 USB, SD카드, CD 등 저장장치에서 원본의 변경없이 데이터를 수집하고, 저장장치의 물리적 훼손, 고의 삭제, 포멧등으로 유실된 데이터를 복원하는 포렌식 분야이다.

모바일 포렌식 : 스마트폰과 같은 휴대용 기기에 남겨지는 다양한 정보(통화기록, 카카오톡, 위치정보 등)를 추출하고 분석하는 분야이다.

네트워크 포렌식 : 네트워크 망을 통해 진행되는 공격을 네트워크 공격이라고 하는데, 이 때, 공격 대상, 방식, 도구, 경로, 공격자 등을 파악하기 위한 데이터를 수집하고 분석하는 분야이다.

데이터베이스 포렌식 : 데이터베이스 파일과 트랜젝션로그를 이용하여 전체 자료에서 조사에 필요한 정보를 수집하고 분석하는 분야이다.

서버 포렌식 : 서버 접속 로그, 파일시스템 로그, 각종 서비스 및 어플리케이션 로그등을 수집, 분석하는 분야이다.

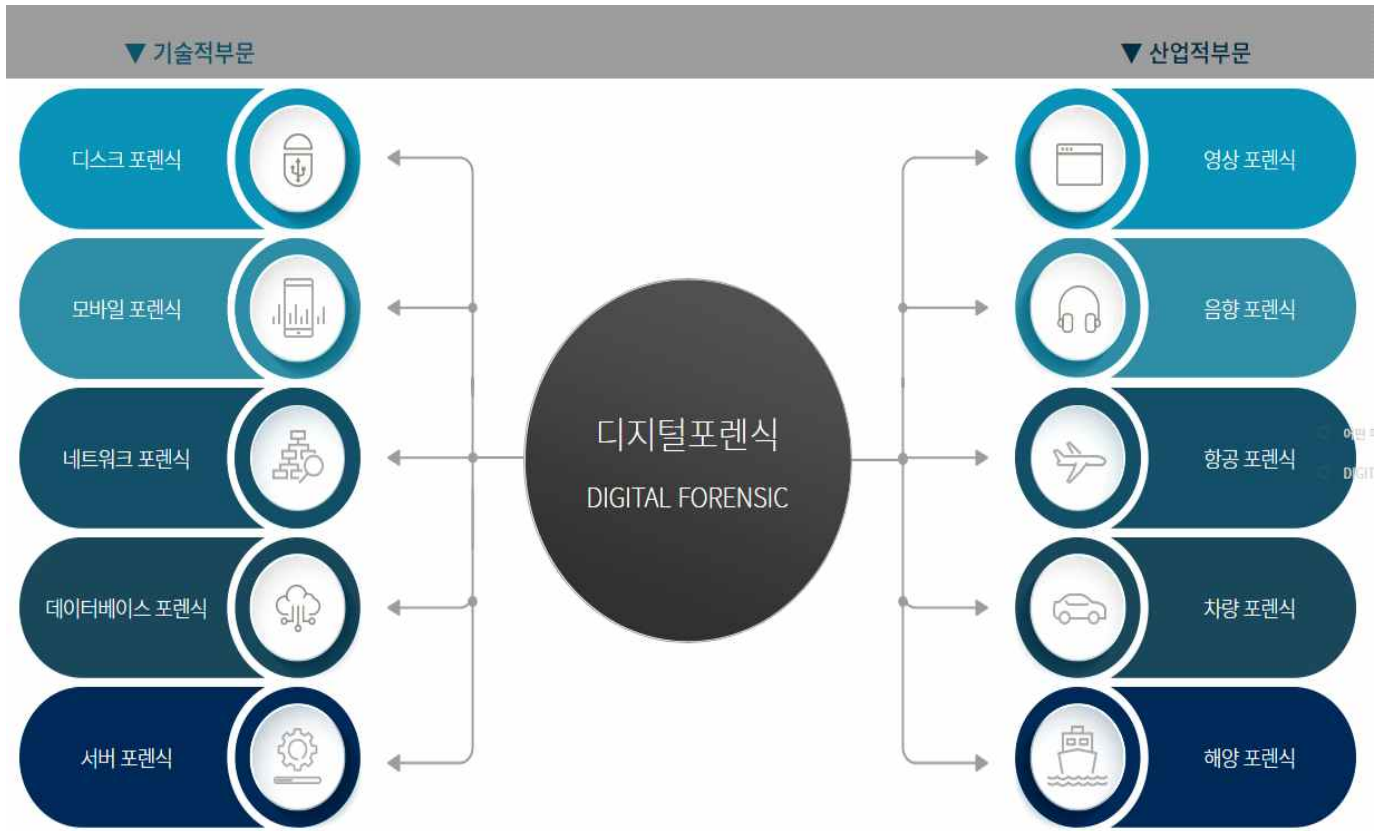
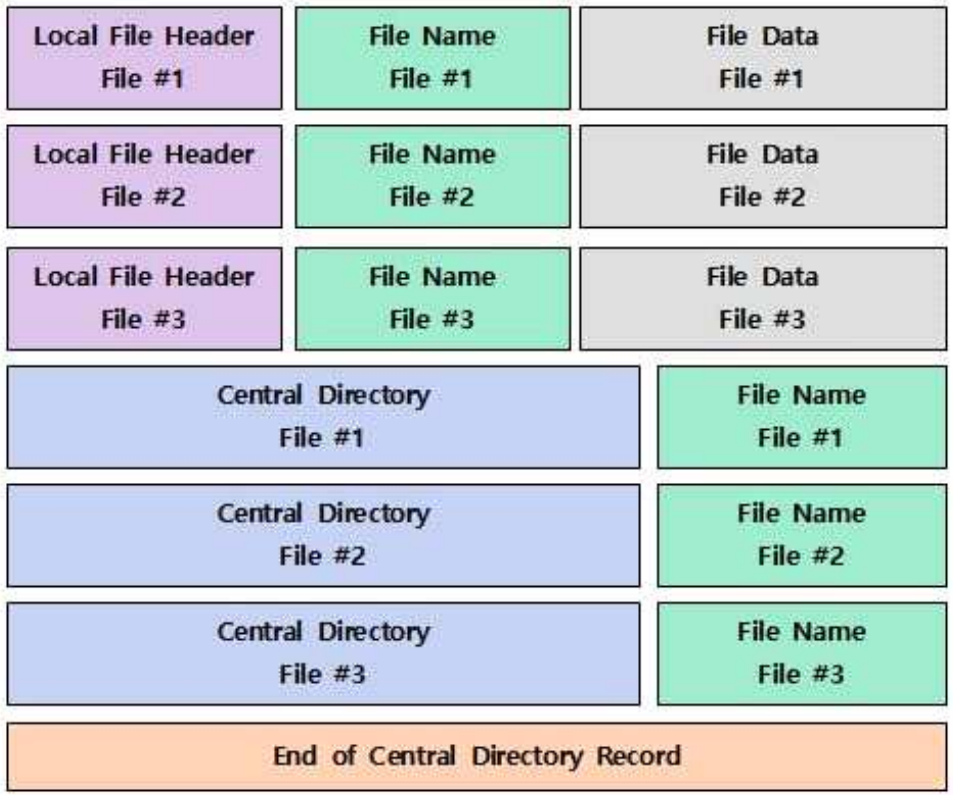


사진 및 자료 출처: 한국디지털포렌식센터 http://www.k-dfc.com/4_1.php

3. ZIP 파일 구조 분석

1) ZIP 파일의 간략한 구조



< ZIP File Simple Structure Layout >

← ZIP 파일 내에 3개의 파일이 압축 되어 있을 때 확인할 수 있는 구조

3. ZIP 파일 구조 분석

2) 준비 과정

math.txt 라는 파일안에 "math is important" 라는 데이터를 작성한 뒤에 zip 파일로 압축을 진행

hxd에 math.zip 넣기

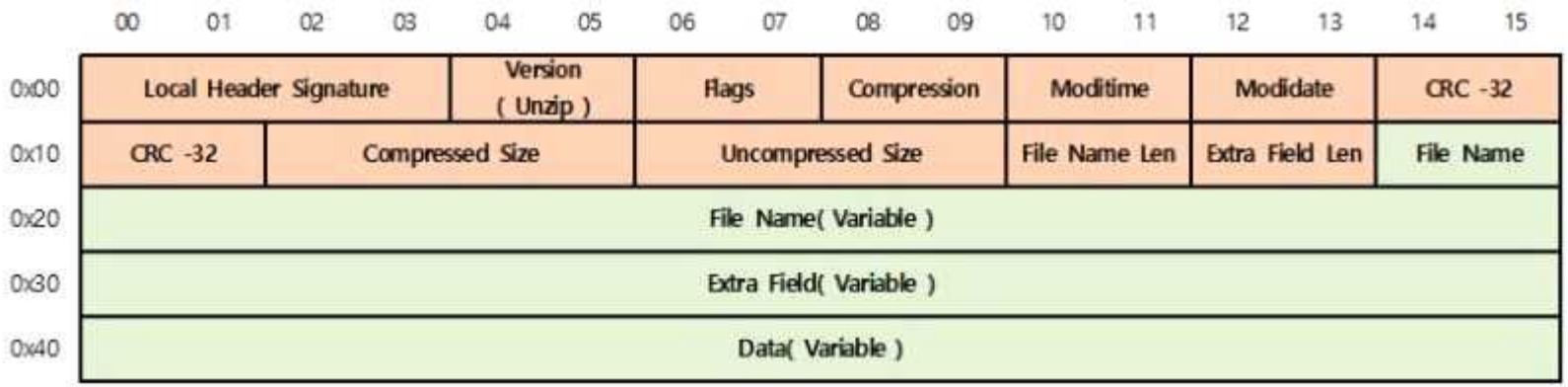
math.zip

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	03	04	14	00	00	00	00	00	72	01	35	52	9D	9F	PK.....r.5R.Ÿ
00000010	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	6D	61	.Ý.....ma
00000020	74	68	2E	74	78	74	6D	61	74	68	20	69	73	20	69	6D	th.txtmath is im
00000030	70	6F	72	74	61	6E	74	2E	50	4B	01	02	14	00	14	00	portant.PK.....
00000040	00	00	00	00	72	01	35	52	9D	9F	0C	DD	12	00	00	00	....r.5R.Ÿ.Ý....
00000050	12	00	00	00	08	00	00	00	00	00	00	00	01	00	20	00	....
00000060	00	00	00	00	00	00	6D	61	74	68	2E	74	78	74	50	4B	......math.txtPK
00000070	05	06	00	00	00	00	01	00	01	00	36	00	00	00	38	00	..........6...8.
00000080	00	00	00	00													....

< math.zip File Full Hex data >

3. ZIP 파일 구조 분석

3) Local File Header



위 구조는 Local File Header의 구조로 고정적인 데이터는 0x00 ~ 0x1D 까지이다.
math.zip 파일의 Hex값을 확인해 보면 아래와 같다.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 03 04 14 00 00 00 00 00 72 01 35 52 9D 9F  PK........r.5R.Ÿ
00000010   0C DD 12 00 00 00 12 00 00 00 08 00 00 00 6D 61  .Ý............ma
00000020   74 68 2E 74 78 74 6D 61 74 68 20 69 73 20 69 6D  th.txtmath is im
00000030   70 6F 72 74 61 6E 74 2E                          portant.
```

3. ZIP 파일 구조 분석

3) Local File Header

3.1) Local File Header - Local Header Signatrue

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 03 04 14 00 00 00 00 00 72 01 35 52 9D 9F  PK........r.5R.Ÿ
00000010   0C DD 12 00 00 00 12 00 00 00 08 00 00 00 6D 61  .Ý............ma
00000020   74 68 2E 74 78 74 6D 61 74 68 20 69 73 20 69 6D  th.txtmath is im
00000030   70 6F 72 74 61 6E 74 2E                          portant.
```

0x00 ~ 0x03은 Local Header Signature이 있는 필드이다. 고정적인 값으로 0x04034B50 (\x50\x4B\x03\x04)이 들어간다.

3.2) Local File Header - Version Required When Decompressing

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 03 04 14 00 00 00 00 00 72 01 35 52 9D 9F  PK........r.5R.Ÿ
00000010   0C DD 12 00 00 00 12 00 00 00 08 00 00 00 6D 61  .Ý............ma
00000020   74 68 2E 74 78 74 6D 61 74 68 20 69 73 20 69 6D  th.txtmath is im
00000030   70 6F 72 74 61 6E 74 2E                          portant.
```

0x04~0x05은 압축해제시에 필요한 버전이 값으로 들어가게 된다.

3) Local File Header

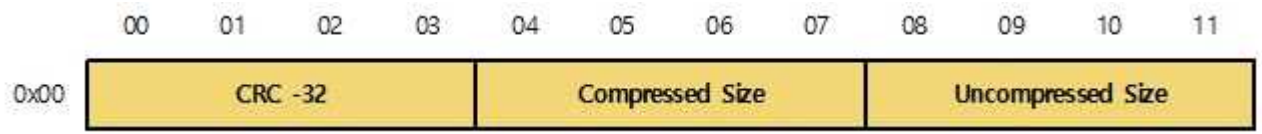
3.3) Local File Header - Flags

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 03 04 14 00 00 00 00 00 72 01 35 52 9D 9F  PK........r.5R.Ÿ
00000010   0C DD 12 00 00 00 12 00 00 00 08 00 00 00 6D 61  .Ý............ma
00000020   74 68 2E 74 78 74 6D 61 74 68 20 69 73 20 69 6D  th.txtmath is im
00000030   70 6F 72 74 61 6E 74 2E                          portant.
```

0x06~0x07은 FLAG 값이 들어 있다.

FLAG에는 어떠한 값이 들어가는지 알아 보면 다음과 같다.

- 0x00 : encrypted file
- 0x01 : compression option
- 0x02 : compression option
- 0x03 : data descriptor, Flags 의 값이 0x03 일때 Data descriptor 라는 게 생성이 되는데 구조는 아래 사진과 같다.
- 0x04 : enhanced deflation
- 0x05 : compressed patched data
- 0x06 : strong encryption
- 0x07-0x10 : unused
- 0x11 : language encoding
- 0x12 : reserved
- 0x13 : mask header values
- 0x14-0x15 : reserved

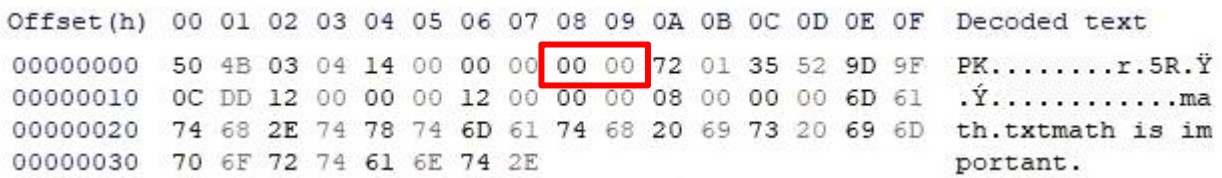


< Local File Header - Data Descriptor >

3. ZIP 파일 구조 분석

3) Local File Header

3.4) Local File Header - Compression Method



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	03	04	14	00	00	00	00	00	72	01	35	52	9D	9F	PK.....r.5R.Ÿ
00000010	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	6D	61	.Ý.....ma
00000020	74	68	2E	74	78	74	6D	61	74	68	20	69	73	20	69	6D	th.txtmath is im
00000030	70	6F	72	74	61	6E	74	2E									portant.

0x08~0x09은 압축 방법이 들어 있다.

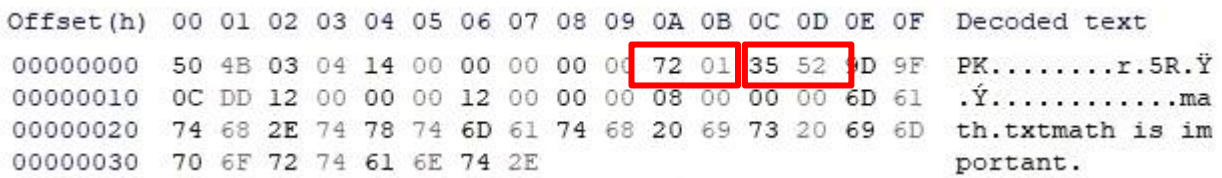
압축 방법으로 어떤 값이 들어 있는지 알아 보면 다음과 같다.

- 0x00 : no compression
- 0x01 : shrunk
- 0x02, 0x03, 0x04, 0x05 : reduced with compression factor 1, 2, 3, 4
- 0x06 : imploded
- 0x07, 0x0B, 0x0D, 0x0F-0x11 : reserved
- 0x08 : deflated
- 0x09 : enhanced deflated
- 0x0A : PKWare DCL imploded
- 0x0C : compressed using BZIP2
- 0x0E : LZMA
- 0x12 : compressed using IBM TERSE
- 0x13 : IBM LZ77 z
- 0x62 : PPMd version I, Rev 1

3. ZIP 파일 구조 분석

3) Local File Header

3.5) Local File Header -File Modification Time/Date



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	03	04	14	00	00	00	00	00	72	01	35	52	9D	9F	PK.....r.5R.Ÿ
00000010	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	6D	61	.Ý.....ma
00000020	74	68	2E	74	78	74	6D	61	74	68	20	69	73	20	69	6D	th.txtmath is im
00000030	70	6F	72	74	61	6E	74	2E									portant.

0x0A~0x0B은 파일 수정 시간을 의미하고 0x0C~0x0D는 파일 수정 날짜를 의미한다.

시간 변환을 해 보면
0x0172 를 2진수로 변환을 해 보면 0x0172 = 0000 0001 0111 0010 이다.
5비트(hhhhh) / 6비트(mmmmmm) / 5비트(sssss)로 나눠 보면 아래와 같고
00000/ 001011/ 10010 이므로 **0시 11분 36초**(초 계산 시 *2) 가 된다.

날짜를 변환해 보면
0x5235 를 2진수로 변환을 해 보면 0x52325 = 0101 0010 0011 0101 이다.
7비트(yyyyyyy) / 4비트(mmmm) / 5비트(ddddd)로 나눠 보면 아래와 같고
0101001/ 0001/ 10101 이므로 **2021년 1월 21일**(년 계산 시 +1980) 가 된다.

3. ZIP 파일 구조 분석

3) Local File Header

3.6) Local File Header - CRC-32 CheckSum

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	03	04	14	00	00	00	00	00	72	01	35	52	9D	9F	PK.....r.5R.Ÿ
00000010	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	6D	61	.Ý.....ma
00000020	74	68	2E	74	78	74	6D	61	74	68	20	69	73	20	69	6D	th.txtmath is im
00000030	70	6F	72	74	61	6E	74	2E									portant.

0x0E~0x11은 CRC-32 알고리즘을 이용해서 file data의 *CRC-32값을 가진다.

* CRC(cyclic redundancy check)는 네트워크 등을 통하여 데이터를 전송할 때 전송된 데이터에 오류가 있는지를 확인하기 위한 체크값을 결정하는 방식

3.7) Local File Header - Compressed Size/Uncompressed Size

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	03	04	14	00	00	00	00	00	72	01	35	52	9D	9F	PK.....r.5R.Ÿ
00000010	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	6D	61	.Ý.....ma
00000020	74	68	2E	74	78	74	6D	61	74	68	20	69	73	20	69	6D	th.txtmath is im
00000030	70	6F	72	74	61	6E	74	2E									portant.

0x12~0x15 / 0x16~0x19 은 압축 크기와 원본 크기가 들어가는 필드이다.

3) Local File Header

3.8) Local File Header - File Name Length/Extra Field Length

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 03 04 14 00 00 00 00 00 72 01 35 52 9D 9F  PK........r.5R.Ÿ
00000010   0C DD 12 00 00 00 12 00 00 00 08 00 00 00 6D 61  .Ý............ma
00000020   74 68 2E 74 78 74 6D 61 74 68 20 69 73 20 69 6D  th.txtmath is im
00000030   70 6F 72 74 61 6E 74 2E                          portant.
```

0x1A~0x1B / 0x1C~0x1D은 파일 이름의 길이와 추가 필드 길이가 들어가는 필드이다.

3.9) Local File Header - File Name

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 03 04 14 00 00 00 00 00 72 01 35 52 9D 9F  PK........r.5R.Ÿ
00000010   0C DD 12 00 00 00 12 00 00 00 08 00 00 00 6D 61  .Ý............ma
00000020   74 68 2E 74 78 74 6D 61 74 68 20 69 73 20 69 6D  th.txtmath is im
00000030   70 6F 72 74 61 6E 74 2E                          portant.
```

0x1E~(0x1E+파일의길이) 는 파일 명이 들어가는 필드이다.

3. ZIP 파일 구조 분석

3) Local File Header

3.10) Local File Header - Extra Field

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	03	04	14	00	00	00	00	00	72	01	35	52	9D	9F	PK.....r.5R.Ÿ
00000010	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	6D	61	.Ý.....ma
00000020	74	68	2E	74	78	74	6D	61	74	68	20	69	73	20	69	6D	th.txtmath is im
00000030	70	6F	72	74	61	6E	74	2E									portant.

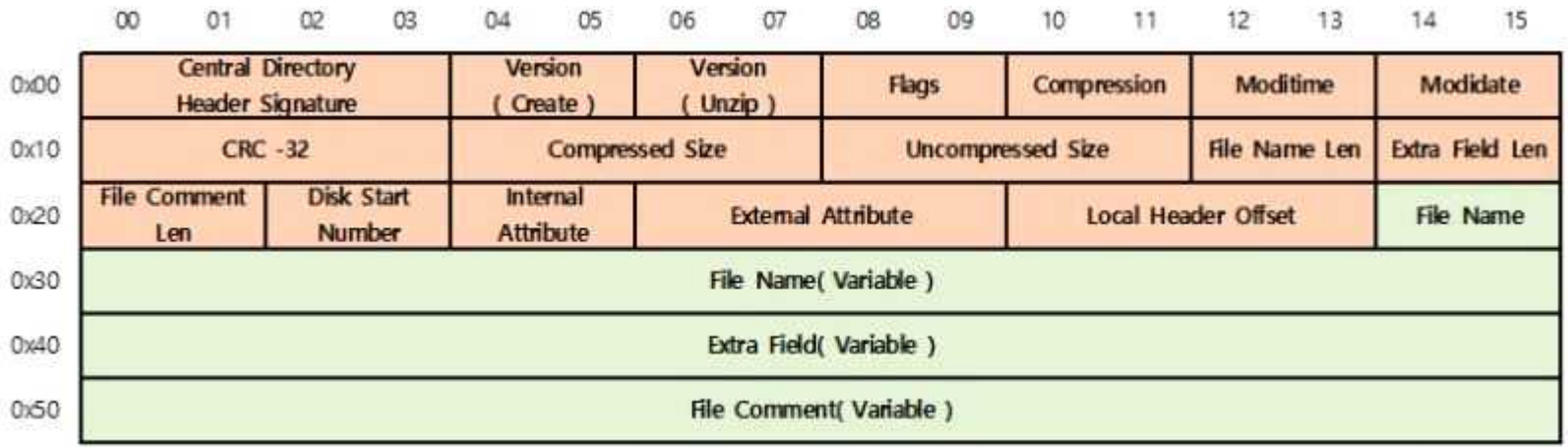
파일 명이 끝난뒤 부터는 추가 필드가 나오게 되며 추가 필드 길이만큼 데이터가 쓰인다.

txt 파일의 크기가 일정 크기 이상이면 아래와 같이 txt 내용 그대로 hxd에 표시 되지 않는다.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	03	04	14	00	00	00	08	00	3C	87	35	52	8F	F7	PK.....<‡5R.÷
00000010	EA	CE	14	00	00	00	59	00	00	00	0C	00	00	00	6C	6F	êÎ....Y.....lo
00000020	6E	67	74	65	78	74	2E	74	78	74	CB	54	C8	48	2C	49	ngtext.txtËTÈH,I
00000030	55	48	CE	C8	CC	4B	2D	4E	55	C8	A4	1A	17	00			UHÎÈÌK-NUÈ¤...

3. ZIP 파일 구조 분석

4) Central Directory File Header



위 구조는 Central Directory File Header의 구조로 고정적인 데이터는 0x00 ~ 0x2D 까지이다.
math.zip 파일의 Hex값을 확인해 보면 아래와 같다.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 01 02 14 00 14 00 00 00 00 00 72 01 35 52  PK..........r.5R
00000010   9D 9F 0C DD 12 00 00 00 12 00 00 00 08 00 00 00  .Ÿ.Ý............
00000020   00 00 00 00 01 00 20 00 00 00 00 00 00 00 6D 61  ...... .......ma
00000030   74 68 2E 74 78 74                                th.txt
```

3. ZIP 파일 구조 분석

4) Central Directory File Header

4.1) Central Directory File Header - Central Directory Header Signatrue

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 01 02 14 00 14 00 00 00 00 00 72 01 35 52  PK..........r.5R
00000010   9D 9F 0C DD 12 00 00 00 12 00 00 00 08 00 00 00  .Ÿ.Ý............
00000020   00 00 00 00 01 00 20 00 00 00 00 00 00 00 6D 61  ...... .......ma
00000030   74 68 2E 74 78 74                                th.txt
```

0x00 ~ 0x03은 Central Directory Header Signature이 있는 필드이다. 고정적인 값으로 0x02014B50(\x50\x4B\x01\x02)이 들어간다.

4.2) Central Directory File Header - Version Required When Creating

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 01 02 14 00 14 00 00 00 00 00 72 01 35 52  PK..........r.5R
00000010   9D 9F 0C DD 12 00 00 00 12 00 00 00 08 00 00 00  .Ÿ.Ý............
00000020   00 00 00 00 01 00 20 00 00 00 00 00 00 00 6D 61  ...... .......ma
00000030   74 68 2E 74 78 74                                th.txt
```

0x04~0x05는 Version은 압축 시에 사용한 압축버전 필드 입니다.

해당 필드에 들어가는 값은 다음 페이지에서 다루겠다.

3. ZIP 파일 구조 분석

Upper Byte :

- 0x00 - MS-DOS and OS/2 (FAT / VFAT / FAT32 file systems)
- 0x01 - Amiga
- 0x02 - OpenVMS
- 0x03 - UNIX
- 0x04 - VM/CMS
- 0x05 - Atari ST
- 0x06 - OS/2 H.P.F.S.
- 0x07 - Macintosh
- 0x08 - Z-System
- 0x09 - CP/M
- 0x0A - Windows NTFS
- 0x0B - MVS (OS/390 - Z/OS)
- 0x0C - VSE
- 0x0D - Acorn Risc
- 0x0E - VFAT
- 0x0F - alternate MVS
- 0x10 - BeOS
- 0x11 - Tandem
- 0x12 - OS/400
- 0x13 - OS/X (Darwin)
- 0x24 - 0xFF : unused

Lower Byte : zip specification version

※ math.zip에서는 0x0014이므로 upper byte는 0x00 lower byte는 0x14이다. 따라서 MS-DOS 2.0 버전이다.

3. ZIP 파일 구조 분석

4) Central Directory File Header

4.3) Central Directory File Header - Version Required When Decompressing

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 01 02 14 00 14 00 00 00 00 00 72 01 35 52  PK..........r.5R
00000010   9D 9F 0C DD 12 00 00 00 12 00 00 00 08 00 00 00  .Ÿ.Ý............
00000020   00 00 00 00 01 00 20 00 00 00 00 00 00 00 6D 61  ...... .......ma
00000030   74 68 2E 74 78 74                                th.txt
```

0x06~0x07은 압축해제 시에 필요한 버전이 값으로 들어가게 된다.

4.4) Central Directory File Header -Flags

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 01 02 14 00 14 00 00 00 00 00 72 01 35 52  PK..........r.5R
00000010   9D 9F 0C DD 12 00 00 00 12 00 00 00 08 00 00 00  .Ÿ.Ý............
00000020   00 00 00 00 01 00 20 00 00 00 00 00 00 00 6D 61  ...... .......ma
00000030   74 68 2E 74 78 74                                th.txt
```

0x08~0x09은 FLAG 값이 들어 있다.

FLAG에 들어가는 값은 오른쪽과 같다..

0x00 : encrypted file
0x01 : compression option
0x02 : compression option
0x03 : data descriptor
0x04 : enhanced deflation
0x05 : compressed patched data
0x06 : strong encryption
0x07-0x0A : unused
0x0B : language encoding
0x0C : reserved
0x0D : mask header values
0x0E-0x0F : reserved

4) Central Directory File Header

4.5) Central Directory File Header - Compression Method

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x0A~0x0B은 압축 방법이 들어 있다.

압축 방법으로 어떤 값이 들어 있는지 알아 보면 다음과 같다. (Local File Header - Compression Method와 동일)

- 0x00 : no compression
- 0x01 : shrunk
- 0x02, 0x03, 0x04, 0x05 : reduced with compression factor 1, 2, 3, 4
- 0x06 : imploded
- 0x07, 0x0B, 0x0D, 0x0F-0x11 : reserved
- 0x08 : deflated
- 0x09 : enhanced deflated
- 0x0A : PKWare DCL imploded
- 0x0C : compressed using BZIP2
- 0x0E : LZMA
- 0x12 : compressed using IBM TERSE
- 0x13 : IBM LZ77 z
- 0x62 : PPMd version I, Rev 1

4) Central Directory File Header

4.6) Central Directory File Header - File Modification Time/Date

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x0A~0x0B은 파일 수정 시간을 의미하고 0x0C~0x0D는 파일 수정 날짜를 의미한다.

시간 변환을 해 보면
0x0172 를 2진수로 변환을 해 보면 0x0172 = 0000 0001 0111 0010 이다.
5비트(hhhhh) / 6비트(mmmmmm) / 5비트(sssss)로 나눠 보면 아래와 같고
00000/ 001011/ 10010 이므로 **0시 11분 36초**(초 계산 시 *2) 가 된다.

날짜를 변환해 보면
0x5235 를 2진수로 변환을 해 보면 0x52325 = 0101 0010 0011 0101 이다.
7비트(yyyyyyy) / 4비트(mmmm) / 5비트(ddddd)로 나눠 보면 아래와 같고
0101001/ 0001/ 10101 이므로 **2021년 1월 21일**(년 계산 시 +1980) 가 된다.

→ Local File Header -File Modification Time/Date와 동일

4) Central Directory File Header

4.7) Central Directory File Header - CRC-32 CheckSum

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x0E~0x11은 CRC-32 알고리즘을 이용해서 file data의 CRC-32값을 가진다.

4.8) Central Directory File Header - Compressed Size/Uncompressed Size

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x12~0x15 / 0x16~0x19 은 압축 크기와 원본 크기가 들어가는 필드이다.

4) Central Directory File Header

4.9) Central Directory File Header - File Name Length/Extra Field Length

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x1A~0x1B / 0x1C~0x1D은 파일 이름의 길이와 추가 필드 길이가 들어가는 필드이다.

4.10) Central Directory File Header - File Comment Length

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x20~0x21 은 파일의 주석 길이를 의미한다.

4) Central Directory File Header

4.11) Central Directory File Header - Disk Start Number

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x22~0x23은 파일이 존재 하는 디스크의 번호를 의미합니다.

4.12) Central Directory File Header - Internal Attribute

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x24~0x25 는 파일의 내부 속성 값을 의미 한다. 해당 필드에 들어가는 값은 아래의 의미를 가진다.

Internal File Attribute :
- 0x00 : Apparent ASCII/text file
- 0x01 : Reserved
- 0x02 : Control Field Records Precede Logical Records
- 0x03~0x10 : Unused.

4) Central Directory File Header

4.13) Central Directory File Header - External Attribute

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x26~0x29 는 파일의 외부 속성 값을 의미한다. 호스트 시스템이 의존 하는 속성 값이 들어간다.

4.14) Central Directory File Header - Local Header Offset

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x2A~0x2D 는 Local Header의 시작 Offset 값이 들어가게 된다.

4) Central Directory File Header

4.15) Central Directory File Header - File Name

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	01	02	14	00	14	00	00	00	00	00	72	01	35	52	PK.....r.5R
00000010	9D	9F	0C	DD	12	00	00	00	12	00	00	00	08	00	00	00	.Ÿ.Ý.....
00000020	00	00	00	00	01	00	20	00	00	00	00	00	00	00	6D	61	....ma
00000030	74	68	2E	74	78	74											th.txt

0x2E~(0x2E+파일의길이) 는 파일 명이 들어가는 필드 이다.

4.16) Central Directory File Header - Extra Field

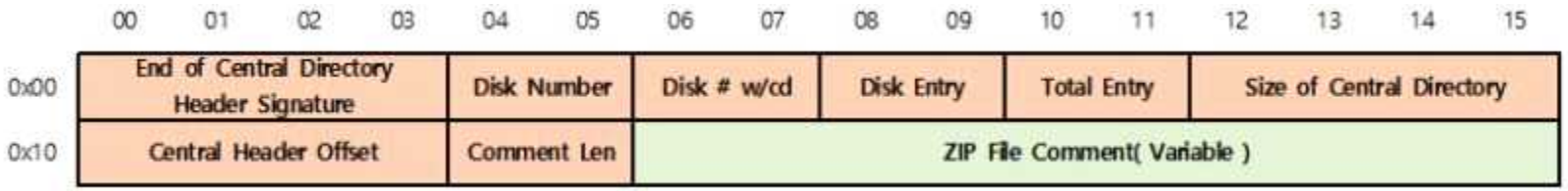
예시 파일에는 Extra Field의 크기가 0이기 때문에 나와 있지 않지만 파일 명이 끝난뒤 부터 추가 필드가 나오게 되며 추가 필드 길이만큼 데이터가 쓰인다.

4.17) Central Directory File Header - File Comment

예시 파일에는 파일 주석의 길이가 0이기 때문에 나와 있지 않지만 추가필드 이후에 나오게 되며 길이는 파일주석의 길이가 적힌 0x20~0x21에 다르다.

3. ZIP 파일 구조 분석

5) End of Central Directory Record

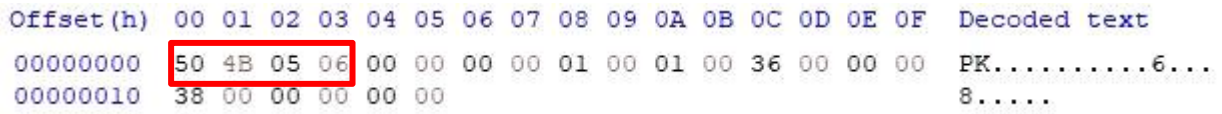


위 구조는 End of Central Directory Record의 구조로 고정적인 데이터는 0x00 ~ 0x15 까지이다.
math.zip 파일의 Hex값을 확인해 보면 아래와 같다.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 05 06 00 00 00 00 01 00 01 00 36 00 00 00  PK..........6...
00000010   38 00 00 00 00 00                                8.....
```

5) End of Central Directory Record

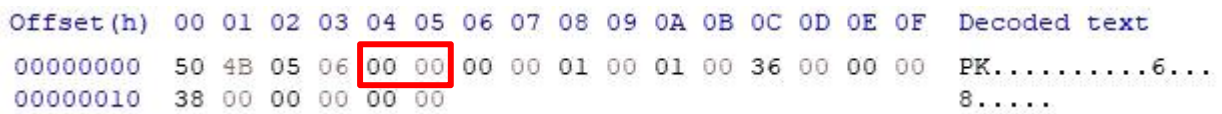
5.1) End of Central Directory Record - End of Central Directory Record Signatrue



```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 05 06 00 00 00 00 01 00 01 00 36 00 00 00  PK..........6...
00000010   38 00 00 00 00 00                                8.....
```

0x00 ~ 0x03은 End of Central Directory Record Signature이 있는 필드이다. 고정적인 값으로 0x06054B50(\x50\x4B\x05\x06)이 들어간다.

5.2) End of Central Directory Record - Disk Start Number



```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 05 06 00 00 00 00 01 00 01 00 36 00 00 00  PK..........6...
00000010   38 00 00 00 00 00                                8.....
```

0x04~0x05은 파일이 존재 하는 디스크의 번호를 의미한다.

5) End of Central Directory Record

5.3) End of Central Directory Record - Disk # w/cd

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 05 06 00 00 00 00 01 00 01 00 36 00 00 00  PK..........6...
00000010   38 00 00 00 00 00                                8.....
```

0x06~0x07은 중앙 디렉토리가 시작되는 디스크 수를 의미한다.

5.4) End of Central Directory Record - Disk Entry

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 05 06 00 00 00 00 01 00 01 00 36 00 00 00  PK..........6...
00000010   38 00 00 00 00 00                                8.....
```

0x08~0x09는 해당 디스크의 Central Directory의 개수를 의미한다.

3. ZIP 파일 구조 분석

5) End of Central Directory Record

5.5) End of Central Directory Record - Total Entry

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 05 06 00 00 00 00 01 00 01 00 36 00 00 00  PK..........6...
00000010   38 00 00 00 00 00                                8.....
```

0x0A~0x0B는 Central Directory의 개수를 의미한다.

5.6) End of Central Directory Record - Size of Central Directory

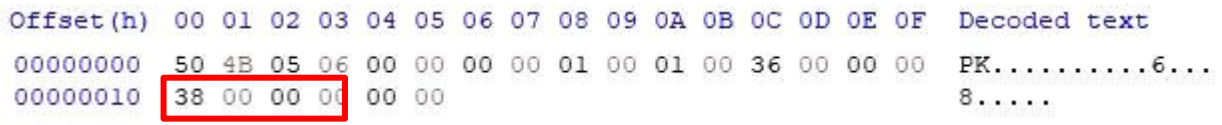
```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   50 4B 05 06 00 00 00 00 01 00 01 00 36 00 00 00  PK..........6...
00000010   38 00 00 00 00 00                                8.....
```

0x0C~0x0F는 Central Directory의 크기를 의미한다.

3. ZIP 파일 구조 분석

5) End of Central Directory Record

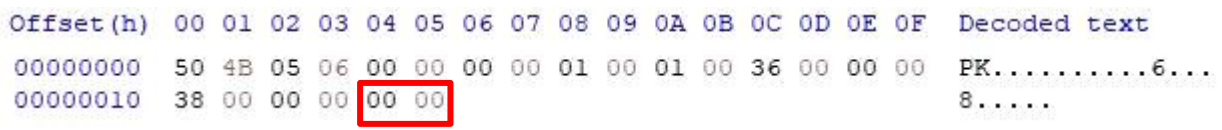
5.7) End of Central Directory Record - Central Header Offset



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	05	06	00	00	00	00	01	00	01	00	36	00	00	00	PK.....6...
00000010	38	00	00	00	00	00											8.....

0x10~0x13은 Central Directory의 시작위치를 의미한다.

5.8) End of Central Directory Record - Comment Length



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	50	4B	05	06	00	00	00	00	01	00	01	00	36	00	00	00	PK.....6...
00000010	38	00	00	00	00	00											8.....

0x14~0x15는 뒤에올 ZIP File Comment의 길이를 의미한다.

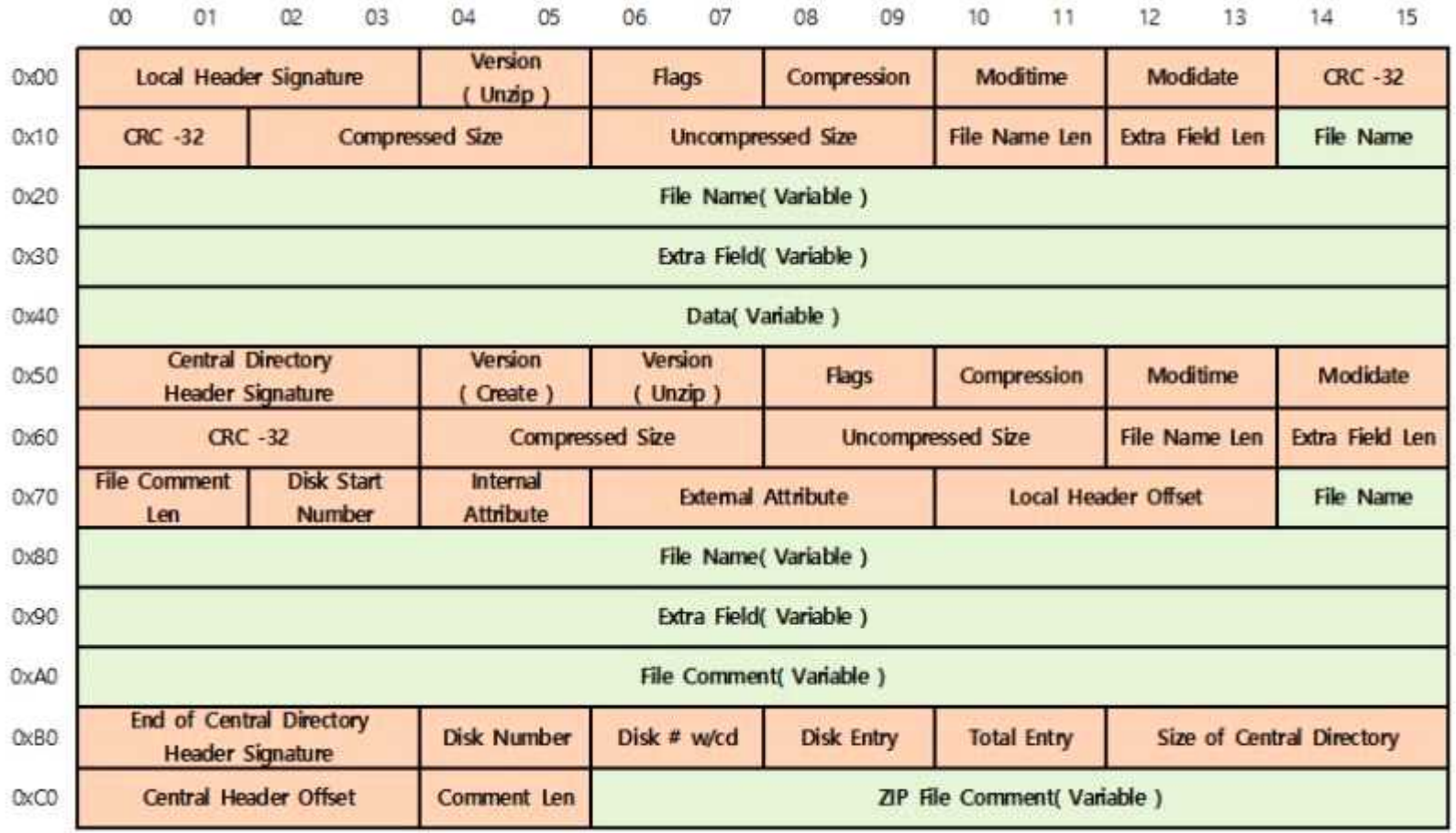
5.9) End of Central Directory Record - ZIP File Comment

해당 예제 파일에서는 Comment Length의 길이가 0이기 때문에 없다.

3. ZIP 파일 구조 분석

6) ZIP file structure

위의 구조들을 보기 쉽게 한번에 나열하면 아래 사진과 같다.



	00 01	02 03	04 05	06 07	08 09	10 11	12 13	14 15
0x00	Local Header Signature		Version (Unzip)	Flags	Compression	Moditime	Modidate	CRC -32
0x10	CRC -32		Compressed Size		Uncompressed Size		File Name Len	Extra Field Len
0x20	File Name(Variable)							File Name
0x30	Extra Field(Variable)							
0x40	Data(Variable)							
0x50	Central Directory Header Signature		Version (Create)	Version (Unzip)	Flags	Compression	Moditime	Modidate
0x60	CRC -32		Compressed Size		Uncompressed Size		File Name Len	Extra Field Len
0x70	File Comment Len	Disk Start Number	Internal Attribute	External Attribute		Local Header Offset		File Name
0x80	File Name(Variable)							
0x90	Extra Field(Variable)							
0xA0	File Comment(Variable)							
0xB0	End of Central Directory Header Signature		Disk Number	Disk # w/cd	Disk Entry	Total Entry	Size of Central Directory	
0xC0	Central Header Offset		Comment Len	ZIP File Comment(Variable)				

< ZIP File Structure full Layout >

사진 및 자료 출처: https://blog.forensicresearch.kr/m/3

4. ZIP 파일 파싱 프로그램 생성

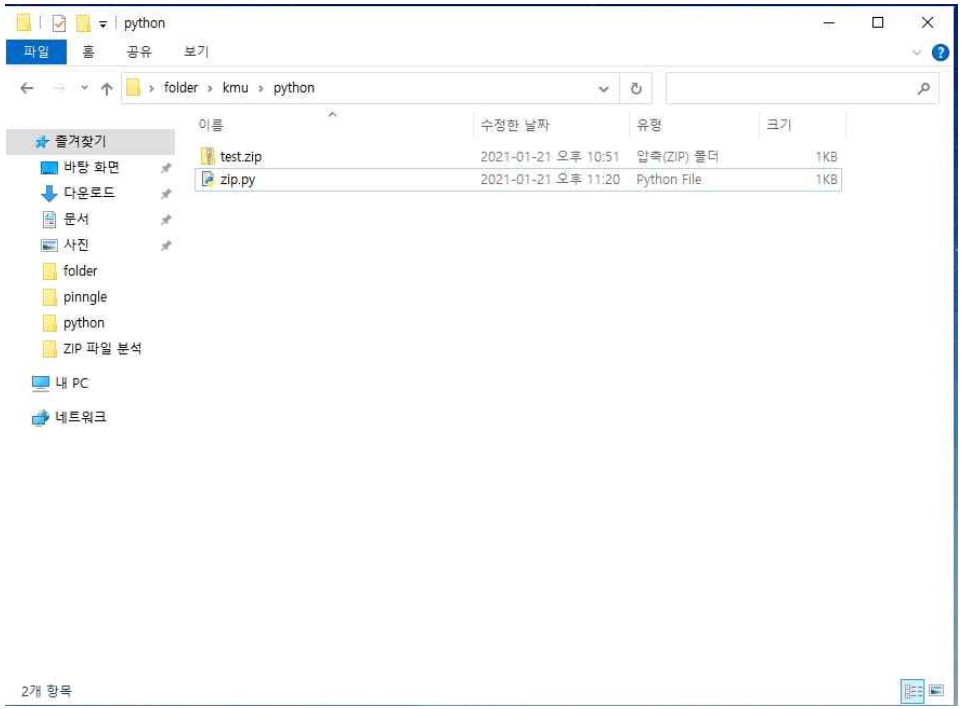
```python
# importing required modules
from zipfile import ZipFile

# specifying the zip file name
file_name = "test.zip"

# opening the zip file in READ mode
with ZipFile(file_name, 'r') as zip:
    # printing all the contents of the zip file
    zip.printdir()

    # extracting all the files
    zip.extractall()

    data = zip.read(test.zip)
```

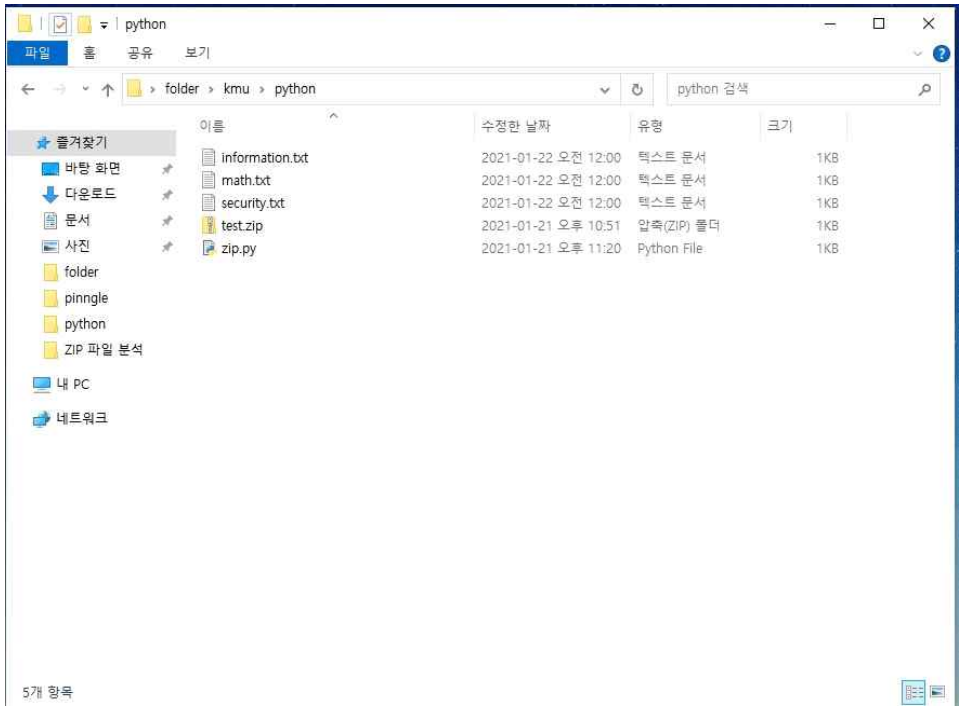
파이썬에서 위와 같이 코드를 짜고 해당 소스 코드가 있는 파일에
information.txt, math.txt, security.txt를 압축한 파일을 넣고 돌려보면 아래와 같은 결과가 나온다

```
File Name                                             Modified             Size
information.txt                                2021-01-21 22:51:24           25
math.txt                                       2021-01-21 00:11:36           18
security.txt                                   2021-01-21 22:51:06           22
```

1) ZIP 파일을 입력으로 넣어서 내부에 있는 파일명 출력 됨

4. ZIP 파일 파싱 프로그램 생성





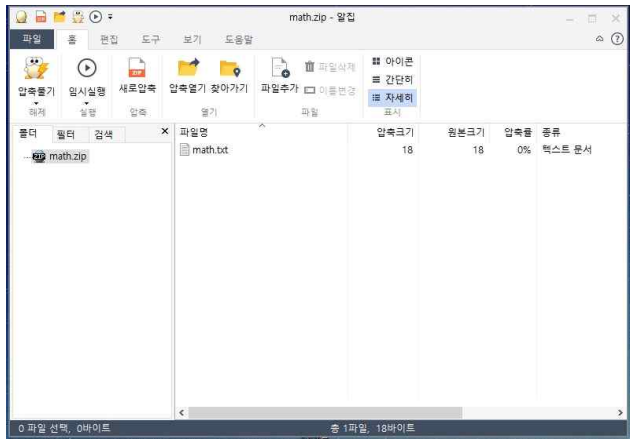
2) 압축이 해제되어 txt파일에 데이터가 저장 됨

3) 파일 데이터가 시작하는offset 출력은?

해당 내용을 구현하지 못 했지만 0x1E~(0x1E+파일의길이) 는 파일 명이 들어가는 필드이고 이 직후 파일 데이터가 시작하는 offset이 나오므로 앞에서 봤던 프로그래밍 결과 값 중에 파일의 크기가 나오므로 이를 이용하여 프로그래밍을 해야 되는 것으로 추측 됨.

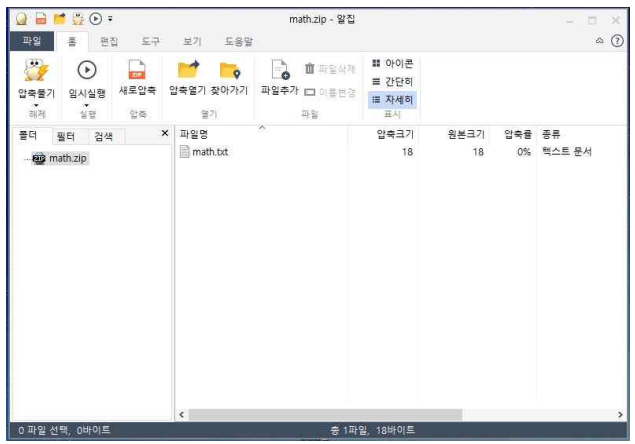
5. 압축 프로그램 사용 시 목록이 안 보이게 하기

Idea 1. End of central directory record 부분의 Total Entry부분이 central directory의 수를 나타내기 때문에 이부분을 0개로 바꾸면 안의 내부 리스트가 안 보일 것이다.

결과 : 

사진과 같이 목록이 보임.

Idea 2. Central Directory File Header - Internal Attribute은 파일 내부의 속성 값을 의미 하기 때문에 내부속성을 변경하면 목록이 안 보일 것이다.

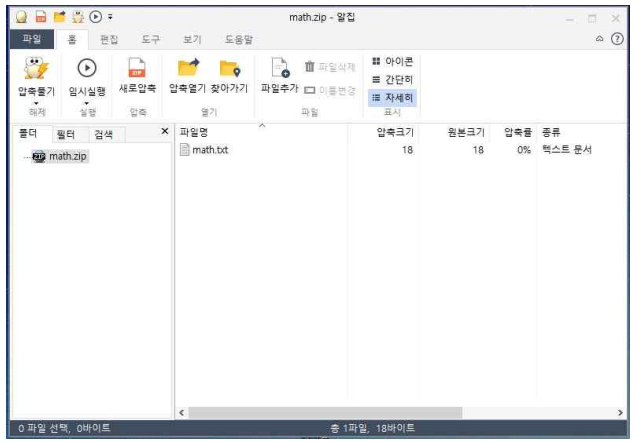
결과 : 

사진과 같이 목록이 보임.

5. 압축 프로그램 사용 시 목록이 안 보이게 하기

Idea 3. Local File Header나 Central Directory File Header에 있는 압축 방법을 나타내는 부분을 변경하면 압축 프로그램 사용 시 목록이 안 보일 것이다.

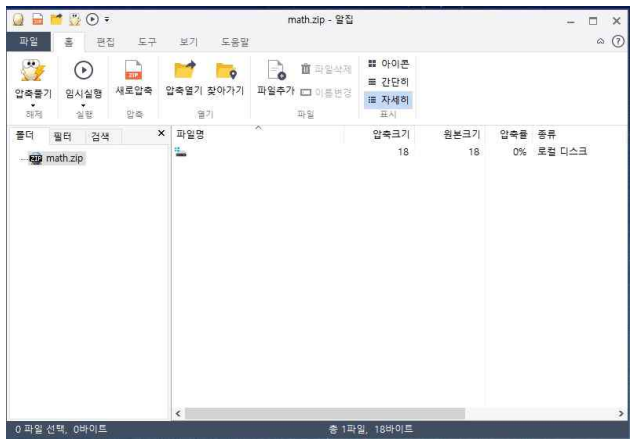
결과 :



사진과 같이 목록이 보임.

Idea 4. Local File Header나 Central Directory File Header에 있는 파일 명을 나타내는 부분에서 파일 명을 없애면 압축프로그램 사용 시 목록이 안 보일 것이다.

결과 :



사진과 같이 종류가 텍스트 문서에서 로컬 디스크로 변경 됨. math.text 라는 목록은 안보이나 맞는 방법인지는 모르겠음.

감사합니다