큰 수 읽기: readhannum

Nova de Hi

2025년 6월 13일 version 0.1

차 례

1 개요 1
1.1 패키지 옵션 1
1.2 명령과 용례 1
2 구현 3
2.1 패키지 옵션 3
2.2 기본 명령 \readbn 3
2.3 문서 명령 \readbnsimple 10
2.4 "만 단위 수사"의 수식 11
2.5 세(네) 자리마다 콤마 찍기 11
2.6 한자로 읽기 12
3 표기 관련 코멘트 12

1 개요

수는 "이만체진" 방법으로 읽는다. 사용되는 단위 수사와 그 값은 다음과 같다.

만(萬)	10^{4}	억(億)	10^{8}	조(兆)	10^{12}
경(京)	10^{16}	해(垓)	10^{20}	자(秭)	10^{24}
양(穰)	10^{28}	구(溝)	10^{32}	간(澗)	10^{36}
정(正)	10^{40}	재(載)	10^{44}	극(極)	10^{48}
항하사(恒河沙)	10^{52}	아승기(阿僧祇)	10^{56}	나유타(那由他)	10^{60}
불가사의(不可思議)	10^{64}	무량대수(無量大數)	10^{68}		

1.1 패키지 옵션

`[hanja]` 옵션이 주어지면 숫자 표기를 한자로 한다. 디폴트는 `hanja=false`이다.

1.2 명령과 용례

다음 명령을 제공한다.

- `\readbn`
- `\readbnsimple`

- \numwithcomma
- \numwithbar

명령의 사용례를 보인다. (예시 숫자에 '경'에 해당하는 부분이 0이므로 이 마디가 나타나지 않아야 한다.)

\readbn 인자로 숫자나 매크로가 올 수 있다.

```
\readbn{12345326530001430240456000094968392939210 98}

\def\mynum{123453265300014302}
\readbn{\mynum}
```

백이십삼정 사천오백삼십이간 육천오백삼십구 십사양 삼천이십사자 사백오십육해 구천사백구십육조 팔천삼백구십이억 구천삼백구십이만 천구십팔
십이경 삼천사백오십삼조 이천육백오십삼억 일만 사천삼백이

\readbnsimple 숫자와 "만 단위 수사"를 섞어써서 표현한다.

```
\readbnsimple{12345326530001430240456000094968392939210 98}

\def\mynum{123453265300014302}
\readbnsimple{\mynum}
```

123정 4532간 6530구 14양 3024자 456해 9496조 8392억 9392만 1098
12경 3453조 2653억 1만 4302

\marknumunit \readbn에서 "만 단위 수사"를 수식한다. 현재 버전에서는 \textsf와 같은 NFSS 폰트 명령을 쓸 수 없는 한계가 있다. 추후 수정할 사항이다. \color나 \underline 등의 명령은 지정할 수 있다. (유니코드 텍 엔진이라면) 직접 \fontspec 관련 명령을 써서 폰트를 바꿀 수 있다.

```
\renewcommand*{\marknumunit}[1]{\textcolor{red}
↪  {\hangulfontspec{HANDotumB-LVT.ttf}#1}}
\readbn{12345324530001430240456000094968392939210 98}
```

백이십삼**정** 사천오백삼십이**간** 사천오백삼십**구** 십사**양** 삼천이십사**자** 사백오십육**해** 구천사백구십육**조** 팔천삼백구십이**억** 구천삼백구십이**만** 천구십팔

\numwithcomma와 \numwithbar 입력된 숫자를 세 자리씩 점을 찍어 표현하거나 네 자리마다 vertical bar를 그어 구분한다.

```
\numwithcomma{12345324530001430240456000094968392939210 98}
```

1,234,532,453,000,143,024,045,600,009,496,839,293,921,098

```
\numwithbar{12345324530001430240456000094968392939210 98}
```

123| 4532| 4530| 0014| 3024| 0456| 0000| 9496| 8392| 9392| 1098

한자

문서 중에서 \hanjaon 또는 \hanjaoff로 한자 선택 기능을 활성화하거나 무력화할 수 있다. 단 한자 폰트는 특히 10^{24}에 해당하는 秭가 인쇄될 수 있는 것이어야 한다. 이 단위를 사용할 정도로 큰 수에서 단위 글자가 인쇄되지 않을 수 있다. pdfTEX에서도 그러하다.

```
\hanjaon
\readbn{12345326530001430240456000094968392939210 98}

\readbnsimple{12345326530001430240456000094968392939210 98}
```

百二十三正 四千五百三十二澗 六千五百三十溝 十四穰 三千二十四秭 四百五十六垓 九千四百九十六兆 八千三百九十二億 九千三百九十二萬 千九十八
123正 4532澗 6530溝 14穰 3024秭 456垓 9496兆 8392億 9392萬 1098

2 구현

```
\ProvidesExplPackage{readhannum}
    {2025/06/13}
    {v0.2}
    {read numbers with hangul}
```

2.1 패키지 옵션

옵션은 [hanja]를 줄 수 있다. l3keys2e 패키지와 keys 자료형을 이용하여 옵션을 처리한다.

```
\RequirePackage{l3keys2e}

\keys_define:nn { readhannum }
{
    hanja    .bool_set:N = \opt_hanja_bool
}

\keys_set:nn { readhannum }
{
    hanja = false
}

\ProcessKeysOptions { readhannum }
```

2.2 기본 명령 \readbn

\readbn은 숫자 인자를 취하여 한글로 읽어 출력한다. 이 패키지의 주요 문서 명령이다. 이를 위한 보조 함수로 다음과 같은 것이 함께 정의될 것이다.

- **\treat_subname:n** '만' 단위의 수사를 붙이기 전에 네 자리 숫자를 읽어서 변환한다. "만 단위 수사"라 함은 **만**, **억**, **조** 등의 자리 수사를 가리킨다. 예를 들면 1234 네 자리 숫자를 **천이백삼십사**라고 읽어주는 것으로 위치에 따라 이 뒤에 "만 단위 수사"가 붙게 된다.

- `\_int_to_hangul:n` 한 자리 숫자의 한글 읽기 변환. 1을 **일**, 2를 **이**, …라고 읽어준다.
- `\_to_outputstr:n` 출력되어야 하는 문자열을 `\l_output_tl`로 보내는 역할을 한다.

1. 출력 문자열을 하나의 매크로에 모았다가 한꺼번에 인쇄하는 방법을 써보려고 한다. 이를 위하여 `tl` 하나를 마련하고 출력 문자열을 이 `tl`에 모으는 함수 `\_to_outputstr:n`을 정의하여둔다. 들어오는 인자를 기본적으로 확장하기로 하자.

```
\tl_new:N \l_outputstr_tl

\cs_new:Npn \_to_outputstr:n #1
{
    \tl_put_right:Nx \l_outputstr_tl { #1 }
}
```

2. 숫자 읽는 단위를 두 개의 `clist`에 넣어둔다. 두 글자 이상의 단위, 즉 **항하사**, **아승기** 등은 한 글자만 (중복되지 않도록) 대표 글자로 넣어두고 나중에 치환할 것임. 한 글자만 넣어두는 이유는 이후 문자열 조작을 쉽게 하기 위해서이다.

   pdfTEX도 지원하기 위하여 "수사"로 치환하는 것을 뒤로 미루고 namespace에는 알파벳 문자를 넣어둔다.

```
\clist_set:Nn \l_namespace_clist {
    a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r
}
\clist_set:Nn \l_subnamespacehangul_clist { 천, 백, 십, 단 }
\clist_set:Nn \l_subnamespacehanja_clist { 千, 百, 十, 단 }
```

3. `\readbn`은 메인 문서명령이다. 인자로 숫자를 받아들여서 한국어 읽기로 변환하여 출력한다.

```
\NewDocumentCommand \readbn { m }
{
```

`\_to_outputstr:n`이 `\tl_put_right:Nn`을 쓰고 있으므로 미리 이 `tl`을 비워두지 않으면 안 된다.

```
    \tl_clear:N \l_outputstr_tl
```

다음 절차는 들어온 인자를 (뒤에서) 네 자리씩 쪼개서 `\l_tmpb_clist`에 넣는 과정이다. `\readbnsimple`과 동일한 과정이므로 이를 별도의 서브루틴으로 분리하였다. `\readbnsimple`에서는 필요없는 처리 절차가 하나 있기 때문에 이를 위하여 `\l_tmpa_bool`을 `true`로 하고 서브루틴을 부른다.

```
    \bool_set_true:N \l_tmpa_bool
    \_subroutine_split:n { #1 }
```

`\l_tmpb_clist`를 `\treat_subname:n` 함수로 mapping하는데 이 함수는 이를테면 12를 **십이**로, 3456을 **삼천사백오십육**으로 읽어 `\l_outputstr_tl`에 모아주는 일을 하게 될 것이다.

```
    \clist_reverse:N \l_tmpb_clist
    \clist_map_function:NN \l_tmpb_clist \treat_subname:n
```

과정이 종료되면 출력해야 할 문자열이 `\l_outputstr_tl`에 모여 있는 상태가 되어 있을 것이다. 마지막으로 문자열의 끝에 `(space)`가 있는지를 검사하여 제거하고 `(space)`를 `\space`로 바꾸어 출력한다. 마지막 `(space)`의 제거는 예컨대 10000을 변환했을 때 "만(space)" 상태가 될 것인데 제거하지 않으면 출력 결과에 스페이스가 하나 따라붙기 때문이다.

```
    \regex_replace_once:nnN { \(space\)$ } { } \l_outputstr_tl
    \regex_replace_all:nnN { \(space\) } { \c{space} } \l_outputstr_tl
    \tl_use:N \l_outputstr_tl
```

여기서 이 문서 명령의 정의는 끝난다.

```
}
```

4. `\_subroutine_split:n`. 이 서브루틴은 숫자를 (뒤에서) 네 자리씩 끊고 "수사"를 붙여 `\l_tmpb_clist`에 저장하는 것이다.

```
\cs_new:Npn \_subroutine_split:n #1
{
```

들어오는 입력열이 매크로일 가능성이 있으므로 일단 확장한 후 처리하도록 하자.

```
    \tl_set:Nx \l_tmpa_tl { #1 }
```

들어온 입력열에서 콤마가 있으면 제거한다. 이렇게 함으로써 콤마로 구분된 숫자도 처리할 수 있게 된다.

```
    \regex_replace_all:nnN { , } {} \l_tmpa_tl
```

만약 들어온 숫자가 $10000 \le N < 20000$이면 `\g_tmpa_bool`을 `true`로 해두기로 한다. 이 처리가 필요한 이유는 10000을 **일만**으로 읽지 않기 위해서이다. 다만 예컨대 **일억 일만**은 **일억 만**으로 읽게 하지 않는다.

숫자가 매우 큰 때가 있으므로 `int`로 크기를 비교할 수 없다. 따라서 문자열의 길이와 첫 문자를 검사하여야 한다. 콤마를 제거한 길이가 5이고 첫 글자가 1이면 `\g_tmpa_bool`을 `true`로 한다.

이 절차는 `\readbnsimple`에서는 필요없다. 그러므로 `\l_tmpa_bool`이 `true`일 때만 실행되도록 한다.

```
    \bool_if:NT { \l_tmpa_bool }
    {
        \bool_if:nTF {
            \int_compare_p:n { \tl_count:N \l_tmpa_tl == 5 }
            &&
            \str_if_eq_p:ee { \str_head:n { #1 } } { 1 }
        }
        { \bool_gset_true:N \g_tmpa_bool }
        { \bool_gset_false:N \g_tmpa_bool }
    }
```

그리고 문자열을 뒤집는다.

```
    \tl_reverse:N \l_tmpa_tl
```

한국어의 숫자 읽기는 네 자리 단위이다. 그러므로 숫자를 네 개마다 한 묶음으로 구분하여야 한다. 네 자리마다 콤마를 추가하는데 이 콤마는 출력에 사용될 것이 아니라 `clist` 또는 `seq`에 저장할 아이템의 구분자이다.

이렇게 한 상태에서 `\l_tmpb_seq`에 이것을 저장한다. 마지막의 콤마를 지웠기 때문에 "빈 아이템"은 생기지 않는다. 예를 들면 인자로 1234560798이 들어왔을 때를 생각하자.
뒤집어서 콤마를 네 자리마다 붙이면 8970,6543,21.
`\l_tmpb_seq`에는 {8970}, {6543}, {21}이라는 세 개의 item이 들어가 있을 것이다.

```
    \regex_replace_all:nnN { (\d\d\d\d) } { \0, } \l_tmpa_tl
    \regex_replace_once:nnN { ,$ } { } \l_tmpa_tl

    \exp_args:NNo \seq_set_from_clist:Nn \l_tmpb_seq { \l_tmpa_tl }
```

이 seq를 indexed mapping할 것인데, 첫번째 루프에서 #1은 1, #2는 8970이 들어온다. 이 숫자를 뒤집으면 0798이 되고 여기에 \l_namespace_clist의 #1째 아이템을 붙이면 0798a라는 문자열로 변한다.

다음번 루프에서는 2와 6543이 들어오는데 같은 절차에 의해 3456b라는 문자열을 만들어 clist에 두번째 아이템으로 저장한다. 단, #2로 들어온 인자(4자리 이하의 수)를 검사해서 이 값이 0이면 아무 것도 하지 않는다. 이런 식으로 진행하면 결국 \l_tmpb_clist는 {0798a},{3456b},{12c}라는 상태가 된다.

이 일을 하려면 루프를 돌리기 전에 \l_tmpb_clist를 비워두어야 한다.

```
    \clist_clear:N \l_tmpb_clist

    \seq_map_indexed_inline:Nn \l_tmpb_seq
    {
        \tl_set:Nf \l_last_tl { \tl_reverse:n { ##2 } }
        \int_compare:nF { ##2 == 0 }
        {
            \tl_put_right:Nx \l_last_tl { \clist_item:Nn \l_namespace_clist { ##1 } }
            \clist_put_right:Nx \l_tmpb_clist { \l_last_tl }
        }
    }
```

이 clist를 뒤집으면 {12c},{3456b},{0798a}이 될 것인데 그럭저럭 숫자 읽기의 형태를 갖추었다.

```
}
```

5. \treat_subname:n의 정의.

```
\cs_new:Npn \treat_subname:n #1
{
```

문자열 12억이 들어온 경우를 생각한다. 이 함수가 하는 일을 대략 스케치하면, 일단 숫자를 무조건 네 자리로 만들어서 처음 자리부터 \l_subnamespace_clist에 있는 순서대로 **천**, **백**, **십**, **단**을 붙여가려는 것이다.

```
    \bool_if:NTF \opt_hanja_bool
    {
        \clist_set_eq:NN \l_subnamespace_clist \l_subnamespacehanja_clist
    }
    {
        \clist_set_eq:NN \l_subnamespace_clist \l_subnamespacehangul_clist
    }
```

그러기 위해서는 일단 숫자를 모두 4자리로 만들어야 한다. 끝에 단위 문자 하나가 붙어 있으므로 4자리 숫자라면 문자열 길이는 5여야 하므로 길이가 5보다 작으면 왼쪽(앞쪽)으로 0을 모자라는 자릿수만큼 붙인다. 그러면 12억은 0012억이 될 것이다.

```
    \tl_set:Nn \l_tmpa_tl { #1 }
    \int_compare:nT { \tl_count:N \l_tmpa_tl < 5 }
```

```
    {
        \int_step_inline:nn { \int_eval:n { 5 - \tl_count:N \l_tmpa_tl } }
        {
            \tl_put_left:Nn \l_tmpa_tl { 0 }
        }
    }
```

여기서 **억**이라는 단위 문자열을 일단 떼어서 `\l_unitname_tl`에 잠시 저장해두고 숫자만 분리해내기로 하자. 다음은 그렇게 하는 과정이다. 단위 문자 없이 숫자만 남은 문자열(0012)은 `\l_tmpb_tl`에 저장되었다.

```
    \tl_reverse:N \l_tmpa_tl
    \tl_set:Nf \l_unitname_tl { \tl_head:N \l_tmpa_tl }
    \tl_set:Nf \l_tmpb_tl { \tl_tail:N \l_tmpa_tl }
    \tl_reverse:N \l_tmpb_tl
```

이제 `\l_tmpb_tl`을 각 토큰에 대하여 mapping하려 하는데 몇 번째 토큰인가를 세기 위하여 `\l_tmpa_int` 정수 변수를 하나 이용한다.

```
    \int_zero:N \l_tmpa_int
```

이 정수 변수가 4인지를 검사하는 것은 현재 처리할 숫자의 자릿수가 **단**의 자리일 때이다. 그러면 자리 이름 **단**을 붙이지 않고 숫자만을 한글로 변환한다. 한 자리 숫자를 한글로 변환하여 `outputstr`에 저장하는 것은 `\_int_to_hangul:n`에 의한다. 이 함수의 정의는 문단 7에 있다.

```
    \tl_map_inline:Nn \l_tmpb_tl
    {
        \int_incr:N \l_tmpa_int
        \int_compare:nTF { \l_tmpa_int == 4 }
        {
            \_int_to_hangul:n { ##1 }
        }
        {
```

정수 변수가 4가 아니라 1이면 **천**, 2면 **백**, 3이면 **십**의 단위를 숫자에 붙여주어야 한다. 만약 들어온 숫자가 0이면 아예 아무 것도 읽지 않고 넘어가고, 1이면 **일**을 새기지 않고 단순히 단위 이름만을 남기도록 처리한다. 그리고 결과를 `\l_outputstr_tl`에 넘기기 위해 `\_to_outputstr:n` 함수를 부른다.

```
            \str_case:nnF { ##1 }
            {
                { 0 } { }
                { 1 } { \_to_outputstr:n { \exp_args:NNf \clist_item:Nn \l_subnamespace_clist { \
            }
            {
                \_int_to_hangul:n { ##1 }
                \_to_outputstr:n { \exp_args:NNf \clist_item:Nn \l_subnamespace_clist { \int_use:N
            }
        }
```

그러면 0012는 최종적으로 0**천**은 생략, 0**백**도 생략, 1**십**은 그냥 **십**, 2**단**은 **이**만 남게 되어 **십이**라는 읽는 방식이 성립하게 되었다.

```
    }
```

이제 여기에 **만**, **억** 등의 단위명을 붙여야 하는데 앞서 `\l_unitname_tl`에 저장해둔 문자열을 알파벳이다. 이것을 인자로 취하여 한글 "만 단위 수사"로 바꾸어주는 서브루틴 `\_attach_names:n`을 호출한다. 이 서브루틴의 정의는 문단 6에 있다.

```
    \_attach_names:n { \l_unitname_tl }
```

십이에 단위 이름 **억**을 붙이면 **십이억**이라는 결과가 되었다. 이것을 출력 문자열에 저장하였다.

여기서 이 함수의 정의가 끝난다.

```
}
```

6. `\_attach_names:n`의 정의. `#1=a`는 **단** 자리인데 아무 것도 적지 않는다.

```
\cs_new:Npn \_attach_names:n #1
{
    \bool_if:NTF \opt_hanja_bool
    {
        \str_case_e:nnT { #1 }
        {
            { a } { \tl_set:Nn \l_thisname_tl {   } }
            { b } { \tl_set:Nn \l_thisname_tl { 萬 } }
            { c } { \tl_set:Nn \l_thisname_tl { 億 } }
            { d } { \tl_set:Nn \l_thisname_tl { 兆 } }
            { e } { \tl_set:Nn \l_thisname_tl { 京 } }
            { f } { \tl_set:Nn \l_thisname_tl { 垓 } }
            { g } { \tl_set:Nn \l_thisname_tl { 秭 } }
            { h } { \tl_set:Nn \l_thisname_tl { 穰 } }
            { i } { \tl_set:Nn \l_thisname_tl { 溝 } }
            { j } { \tl_set:Nn \l_thisname_tl { 澗 } }
            { k } { \tl_set:Nn \l_thisname_tl { 正 } }
            { l } { \tl_set:Nn \l_thisname_tl { 載 } }
            { m } { \tl_set:Nn \l_thisname_tl { 極 } }
            { n } { \tl_set:Nn \l_thisname_tl { 恒河沙 } }
            { o } { \tl_set:Nn \l_thisname_tl { 阿僧祇 } }
            { p } { \tl_set:Nn \l_thisname_tl { 那由他 } }
            { q } { \tl_set:Nn \l_thisname_tl { 不可思議 } }
            { r } { \tl_set:Nn \l_thisname_tl { 無量大數 } }
        }
    }
    {
        \str_case_e:nnT { #1 }
        {
            { a } { \tl_set:Nn \l_thisname_tl {   } }
            { b } { \tl_set:Nn \l_thisname_tl { 만 } }
            { c } { \tl_set:Nn \l_thisname_tl { 억 } }
            { d } { \tl_set:Nn \l_thisname_tl { 조 } }
            { e } { \tl_set:Nn \l_thisname_tl { 경 } }
            { f } { \tl_set:Nn \l_thisname_tl { 해 } }
```

```
            { g } { \tl_set:Nn \l_thisname_tl { 자 } }
            { h } { \tl_set:Nn \l_thisname_tl { 양 } }
            { i } { \tl_set:Nn \l_thisname_tl { 구 } }
            { j } { \tl_set:Nn \l_thisname_tl { 간 } }
            { k } { \tl_set:Nn \l_thisname_tl { 정 } }
            { l } { \tl_set:Nn \l_thisname_tl { 재 } }
            { m } { \tl_set:Nn \l_thisname_tl { 극 } }
            { n } { \tl_set:Nn \l_thisname_tl { 항하사 } }
            { o } { \tl_set:Nn \l_thisname_tl { 아승기 } }
            { p } { \tl_set:Nn \l_thisname_tl { 나유타 } }
            { q } { \tl_set:Nn \l_thisname_tl { 불가사의 } }
            { r } { \tl_set:Nn \l_thisname_tl { 무량대수 } }
        }
    }
    {
        \_to_outputstr:n { \marknumunit { \l_thisname_tl } (space) }
    }
}
```

7. `\_int_to_hangul:n`의 정의.

들어오는 숫자를 한글로 변환하여 돌려준다.

```
\cs_new:Npn \_int_to_hangul:n #1
{
```

한자 옵션이 활성화되어 있으면

```
    \bool_if:NTF \opt_hanja_bool
    {
        \str_case:nn { #1 }
        {
            { 1 } { \_to_outputstr:n { 一}}
            { 2 } { \_to_outputstr:n { 二}}
            { 3 } { \_to_outputstr:n { 三}}
            { 4 } { \_to_outputstr:n { 四}}
            { 5 } { \_to_outputstr:n { 五}}
            { 6 } { \_to_outputstr:n { 六}}
            { 7 } { \_to_outputstr:n { 七}}
            { 8 } { \_to_outputstr:n { 八}}
            { 9 } { \_to_outputstr:n { 九}}
        }
    }
```

한자 옵션이 `false`이면 한글로

```
    {
        \str_case:nn { #1 }
        {
```

5자리 숫자에 첫 숫자가 1(`\g_tmpa_bool = true`)이고 `hanja` 상태가 아니면 **일**을 새기지 않음

```
            { 1 } { \bool_if:NF \g_tmpa_bool { \_to_outputstr:n { 일 } } }
```

그 외에는 평이한 읽기

```
            { 2 } { \_to_outputstr:n { 이 } }
            { 3 } { \_to_outputstr:n { 삼 } }
            { 4 } { \_to_outputstr:n { 사 } }
            { 5 } { \_to_outputstr:n { 오 } }
            { 6 } { \_to_outputstr:n { 육 } }
            { 7 } { \_to_outputstr:n { 칠 } }
            { 8 } { \_to_outputstr:n { 팔 } }
            { 9 } { \_to_outputstr:n { 구 } }
        }
    }
}
```

2.3 문서 명령 \readbnsimple

이 명령은 1,234,565,700과 같은 숫자를 12억 3456만 5700과 같이 출력하는 것이다. 즉 "만 단위 수사"를 제외한 부분은 아라비아 숫자로 그대로 제시한다. 단 이 부분이 0이면 생략하는 것은 같다.

```
\NewDocumentCommand \readbnsimple { m }
{
    \tl_clear:N \l_outputstr_tl
```

\readbn과 달리 \l_tmpa_bool을 false로 하여 서브루틴을 부른다.

```
    \bool_set_false:N \l_tmpa_bool
    \_subroutine_split:n { #1 }
```

이제 넘어오는 \l_tmpb_clist를 뒤집어서 출력하기만 하면 되겠다.

```
    \clist_reverse:N \l_tmpb_clist
    \clist_map_inline:Nn \l_tmpb_clist
    {
```

넘어오는 ##1의 "단위명"과 숫자를 분리한다.

```
        \tl_set:Nn \l_tmpa_tl { ##1 }
        \tl_reverse:N \l_tmpa_tl
        \tl_set:Nf \l_tmpb_tl { \tl_head:N \l_tmpa_tl }
        \tl_set:Nf \l_tmpc_tl { \tl_tail:N \l_tmpa_tl }
        \tl_reverse:N \l_tmpc_tl
```

"단위명"은 \l_tmpb_tl에, 숫자 부분은 \l_tmpc_tl에 들어갔다. 이제 숫자 부분을 출력 문자열로 보내고 _attach_names:n을 호출하여 "단위명"의 한글식 표현을 붙인다. 이 부분에서 \int_eval:n을 쓴 이유는 예컨대 0004양과 같은 경우에 4양으로만 인쇄되게 하기 위한 것이다.

```
        \exp_args:Nf
        \_to_outputstr:n { \exp_args:No \int_eval:n { \l_tmpc_tl } }
        \_attach_names:n { \l_tmpb_tl }
    }
```

마지막에 (space)를 검사하는 것은 동일하다.

```
    \regex_replace_once:nnN { \(space\)$ } { } \l_outputstr_tl
    \regex_replace_all:nnN { \(space\) } { \c{space} } \l_outputstr_tl
    \tl_use:N \l_outputstr_tl
```

여기서 이 문서 명령의 정의는 끝난다.

```
}
```

2.4 "만 단위 수사"의 수식

만 단위 수사를 강조하여 표현하고 싶을 때가 있다. 다음 명령을 이를 위하여 마련되어 있고 애초값은 아무것도 하지 않는 것이다.

```
\providecommand\marknumunit{}
```

2.5 세(네) 자리마다 콤마 찍기

큰 수는 세 단위마다 콤마를 찍거나 네 단위마다 가운뎃점을 찍어 보여주면 좋다. `\numwithcomma`는 세 자리마다 점을 찍는 것이고 `\numwithbar`는 네 자리마다 구분 기호를 찍게 하려 한다.

8. 거의 동일한 두 명령을 정의할 것이므로 일단 다음과 같은 함수를 하나 정의하였다.

```
\cs_new:Npn \_num_with_comma_fn:nnn #1 #2 #3
{
```

`#1`은 들어오는 문자열이다.
`#2`는 `regex` 명령으로 주어지는 `regex` 구문이고 `#3`은 구분자 부호(콤마)이다.

```
    \tl_set:Nx \l_tmpa_tl { #1 }
    \regex_set:Nn \l_tmpa_regex { #2 }
```

들어온 문자열에 콤마가 이미 있으면 일단 제거하고 한번 뒤집는다.

```
    \regex_replace_all:nnN {,} {} \l_tmpa_tl
    \tl_reverse:N \l_tmpa_tl
```

세(네) 자리 숫자마다 콤마(구분부호)를 추가한다. 세 자리라면

```
\regex_replace_all:nnN { (\d\d\d) } { \0, } \l_tmpa_tl
```

로 되는데 자릿수가 3의 배수일 때 끝에 불필요한 콤마가 하나 따라붙을 수 있다. 그러므로 만약 맨끝에 콤마가 오면 제거한다. 세 자리마다 끊을지 네 자리마다 끊을지는 이 함수를 호출할 때의 `#2`에 달려 있다.

```
    \regex_replace_all:NnN \l_tmpa_regex { \0 #3 } \l_tmpa_tl
    \regex_replace_once:nnN { #3$ } {} \l_tmpa_tl
```

다시 뒤집어서 출력

```
    \tl_reverse:N \l_tmpa_tl
    \tl_use:N \l_tmpa_tl
}
```

9. 앞서 정의한 _num_with_comma_fn:nnn을 써서 \numwithbar와 \numwithcomma를 정의한다.

```
\NewDocumentCommand \numwithcomma { m }
{
    \_num_with_comma_fn:nnn { #1 } { (\d\d\d) } { , }
}
```

\numwithbar의 구분자로 \ensuremath{\vert}를 써볼까 한다. 좀 길고 장황하므로 새로운 매크로로 다음과 같이 정의하여 써먹으면 되겠다. 이왕 매크로 정의로 하는 거 색깔도 넣고 해봤다.

```
\tl_set:Nn \l_myvert_tl {\ensuremath{\color{blue}\vert\,}}

\NewDocumentCommand \numwithbar { m }
{
    \_num_with_comma_fn:nnn { #1 } { (\d\d\d\d) } { \c{l_myvert_tl} }
}
```

이 명령이 실행되는 예를 들어 보면 다음과 같다.

\numwithcomma{123450697} 123,450,697

\numwithbar{123450697} 1|2345|0697

2.6 한자로 읽기

한자는 패키지 옵션으로 줄 수도 있고 다음 두 명령을 이용하여 활성화하거나 취소할 수 있다.

```
\NewDocumentCommand \hanjaon {}
{
    \bool_set_true:N \opt_hanja_bool
}

\NewDocumentCommand \hanjaoff {}
{
    \bool_set_false:N \opt_hanja_bool
}
```

3 표기 관련 코멘트

'만'과 '일만'의 경우. \readbnsimple에서는 '1만'을 그대로 새긴다. 한자 상태에서도 '一萬'으로 된다.

```
\readbn{100010000}\quad \readbnsimple{100010000}\\
\readbn{10000}\quad \readbnsimple{10000}\\
\hanjaon
\readbn{10000}\quad \readbnsimple{10000} \\
\hanjaoff
\readbn{10010}\quad \readbnsimple{10010}
```

일억 일만　1억 1만
만　1만
一萬　1萬
만 십　1만 10

조사가 붙을 때의 스페이스

```
\readbn{10020}을\\
\readbn{50000}을\\
\readbn{100010000}과\\
\readbnsimple{10020}을\\
\readbnsimple{50000}을\\
\readbnsimple{100010000}과\\
```

만 이십을
오만을
일억 일만과
1만 20을
5만을
1억 1만과