1 Introduction

This little \LaTeX{} package implements the macro \athnum. The macro transforms an Arabic numeral, i.e., the kind of numerals we all use (e.g., 1, 5, 789 etc), to the corresponding Athenian numeral. Athenian numerals were in use only in ancient Athens. The special multiples, which the system employs, are drawn with the special macro \PiIt. The macro produces a more or less \Pi-like shape above a letter.

2 The Numbering System

The Athenian numbering system, like the Roman one, employs letters to denote important numbers. Multiple occurrence of a letter denote a multiple of the “important” number, e.g., the letter I denotes 1, so III denotes 3. Here are the basic digits used in the Athenian numbering system:

- I denotes the number one (1)
- II denotes the number five (5)
- Δ denotes the number ten (10)
- Η denotes the number one hundred (100)
- X denotes the number one thousand (1000)
- M denotes the number ten thousands (10000)

Moreover, the letters Δ, Η, X, and M under the letter II, denote five times their original value, e.g., the symbol \[\text{II} \times 5\] denotes the number 5000, and the symbol \[\text{X} \times 5\] denotes the number 50. It must be noted that the numbering system does not provide negative numerals or a symbol for zero.

3 The Code

Before we do anything further, we have to identify the package.

\*\begin{verbatim}
\ProvidesPackage{grnumalt}[1997/09/19 \space v1.1]
\typeout{Package: 'grnumalt' v1.0 \space <1997/09/19> (AS)}
\end{verbatim}\*

\PiIt It is very important to be able to correctly typeset the multiples of the numbering system. For this purpose we define the macro \PiIt. The macro uses two “length” variables.

\*\begin{verbatim}
\newdimen\@boxW \newdimen\@boxH
\end{verbatim}\*

We make the \PiIt macro a robust command.

\*\begin{verbatim}
\DeclareRobustCommand{\PiIt}{% In order to correctly produce the \( \Pi \) symbol we need to know the height and width of the letter that goes under a \( \Pi \). This is done by using the standard \LaTeX{} macros: \settowidth and \settoheight.
\settowidth{\@boxW}{#1}\settoheight{\@boxH}{#1} Since, the width of an ordinary rule is 0.4 pt we must add 0.8 pt to the width of the letter.
\addtolength{\@boxW}{0.8pt} Now comes the interesting part: the actual drawing. We create a vertical box. Inside this box we draw a horizontal rule of width equal to the width of the letter. Next, we create a horizontal box in order to make the vertical lines. We draw the first vertical line, then we put the letter in a \mbox, since it may be a mathematical symbol\footnote{Greek letters are considered mathematical symbols by \LaTeX{}}. After the \mbox we draw the second vertical line and we “close” the horizontal box. A little white space is put after the vertical box, so that adjacent multiplies do not look ugly!
\vbox{\hrule width\@boxW\hbox{\vrule height\@boxH\mbox{#1}\vrule height\@boxH}}\kern.5pt}
\end{verbatim}\*

\athnum Now, we turn our attention to the definition of the macro \athnum. This macro uses one integer variable.

\*\begin{verbatim}
\newcount@ath@num
\end{verbatim}\*

The macro \athnum is also defined as a robust command.

\*\begin{verbatim}
\DeclareRobustCommand{\athnum}{% The macro does not work in math mode so we must ensure that it will not be used in math mode. We could use \ensuremath, but our definition is too long...
\ifmmode\errhelp{``J This macro has been defined to work``J *only* in non-math mode. It is definitely``J sure that you are using it in math mode.``J}\errmessage{``JYou can’t use macro atheniannumeral``J in math mode.``J}\else...
\begin{verbatim}
\end{verbatim}\*

\footnote{Greek letters are considered mathematical symbols by \LaTeX{}}
If we are not in math mode, we can start computing the Athenian numeral. After assigning to variable \athnum the value of the macro’s argument, we make sure that the argument is in the expected range, i.e., it is greater than zero. In case it isn’t we simply produce a \space, warn the user about it and quit.

```
\else\athnum#1\relax
\ifnum\athnum=\z@
\PackageWarning{grnumalt}{Illegal value (\the\athnum) for athenian numeral}\
\else\ifnum\athnum<\z@
\PackageWarning{grnumalt}{Illegal value (\the\athnum) for athenian numeral}\
\else$
```

Having done all the necessary checks, we are now ready to do the actual computation. If the number is greater than 49999, then it certainly has at least one \M “digit”. We find all such digits by continuously subtracting 50000 from \NumA, until \NumA becomes less than 50000.

```
\loop\ifnum\athnum>49999
PiIt\{\mathrm{M}\}
\advance\athnum-50000
\repeat
```

We now check for tens of thousands.

```
\loop\ifnum\athnum>9999
\mathrm{M}\advance\athnum-\@M
\repeat
```

Since a number can have only on \X “digit” (equivalent to 5000), it is easy to check it out and produce the corresponding numeral in case it does have one.

```
\ifnum\athnum>4999
PiIt\{\mathrm{X}\}
\advance\athnum-5000
\fi
```

Next, we check for thousands, the same way we checked for tens of thousands.

```
\loop\ifnum\athnum>999
\mathrm{X}\advance\athnum-\@m
\repeat
```

Like the five thousands, a numeral can have at most one \H “digit” (equivalent to 500).

```
\ifnum\athnum>499
PiIt\{\Delta\}
\fi
```

It is time to check hundreds, which follow the same pattern as thousands

```
\loop\ifnum\athnum>99
\mathrm{H}\advance\athnum-100
\repeat
```

A numeral can have only one \D “digit” (equivalent to 50).

```
\ifnum\athnum>49
PiIt\{\Delta\}
```
Let's check now decades.

We check for fives and, finally, for the digits 1, 2, 3, and 4.

The command \@athnum is defined just to make it possible to have, e.g., page numbering with athenian numerals.

Dedication
I would like to dedicate this piece of work to my son Demetrios-Georgios.