Tartalomjegyzék

Newcommand

- With the **\newcommand** command we can define new commands. These should be placed in the preamble (before **\begin{document}**).
- We can think of it as an abbreviation.
- For example, if we write the set of real numbers a lot of times, we can do the following:

```
...
\newcommand{\R}{\mathbb{R}}
...
\begin{document}
\[
x^2 \geq 0 \quad \forall x\in\R
\]
\end{document}
```

- With this **newcommand** if we write **\R** it gets substituted with **\mathbb{R}**.
- The first parameter of **newcommand** is the name of the command (given by us), the second is the existing command (can be a set of commands) we wish to shorten.
- We could do the same for the set of integers, etc.

Arguments

- We can define complex commands, for example if we would like to make a command for a nice integral:

```
...
\newcommand{\myint}[2]{\int #1 \,\mathrm{d}#2}
...
\begin{document}
\[
\myint{x^2 \sin^2 x}{x} + \myint{x^2 \cos^2 x}{x} = \myint{x^2}{x}
\]
```

```
\end{document}
```

- Here the optional parameter **[2]** tells latex that the command has two arguments (inputs).
- We can specify the position of the arguments by **#1**, **#2**... In the example the first parameter is the integrand, the second is the variable.

- If the command that we want to define already exists we will get an error. For example if we tried to name the previous command **int** instead of **myint**.
- If we want to forcefully redefine a command, we can use the **\renewcommand** that has the same syntax, but it allows us to redefine existing commands.

Tasks

Sample latex document frame:

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[magyar]{babel}

\usepackage{amsmath}
\usepackage{amsthm}
\usepackage{amssymb}

\begin{document}



\end{document}
```

Sets

Define the set of racional and natural numbers. Try it on a simple formula.

Vector

Define a command that has one argument. All it should do is to put a right arrow over the given argument (**\overrightarrow{something}**). This makes it a bit easier to write vectors in formulas. Name it **vec** at first, and after we get an error (because the **\vec** command already exists), name it **myvec**. Try it with an arbitrary formula.

In truth this did not shorten our code all that much, since myvect is only 8 letters less. But still this is very useful when we write longer articles. Just think about a situation where we write a 20 page article using **\overrightarrow** to denote vectors, but then we change our mind and want to use **\underline** instead. Now we can use the replace all functionality (present in most text editors), but what if we used **\overrightarrow** somewhere else that's not a vector? It's all a mess now. Now imagine the same situation if we defined a new command for our vectors and used it throughout the document. All we have to change in this case is the definition of our vector command. This is the true value of using our own commands.

Previously

Theorems, definitions

Let's do some theorems. For that append this to the preamble:

```
\newtheorem{mydef}{Definition}
```

- Create a new threorem style environment!
- Try the different styles (remark, theorem, definition)!

Labels, references

Create references to your theorems:

```
\begin{theorem}\label{thm:sample_thm}
Theorem text
\end{theorem}

In Theorem \ref{thm:sample_thm} we...
```

Floating pictures

```
\begin{figure}[p]
    \centering
    \includegraphics[width=0.8\textwidth]{image.png}
    \caption{Awesome Image}
    \label{fig:awesome_image}
\end{figure}
```

Change the placement (h,t,p,b,!,H)!

BibTeX

BibTeX is a package to create nice looking bibliographies. Create a test bibliography using these sites: http://www.bibtex.org/Using/, https://en.wikibooks.org/wiki/LaTeX/Bibliography_Management.