

udstasksheet – Documentation

ERROR: Missing \term

1 Task 1 – Getting Started

The easiest way to use this template is to copy the fitting example code from the `examples/` folder. In there, you will find three different variants for different ways of submitting tasks:

- `all-exercises-in-one-file` The whole source code for each task sheet is packed into a single \LaTeX source file. Recommended if you submit your sheets alone and all tasks should be submitted in one PDF file.
- `exercises-in-split-files` The source code is split into a separate file for each task but all tasks are compiled into a single PDF file. Recommended if you solve sheets in a group (especially when using Git) and all tasks should be submitted in one PDF file.
- `pdf-per-exercise` Both the source code and the PDF output is split by task into separate files. Recommended if task sheets should be submitted with a PDF per task instead per sheet.

Once you picked the right example code, you have to change a few lines to it to be ready for use.

First of all, you should set the language for your submission. To do so, enter the correct argument (`ngerman` or `english`) for the `\documentclass` command.

```
1 \documentclass[10pt,a4paper,ngerman]{article} % For submissions in German
2 \documentclass[10pt,a4paper,english]{article} % For submissions in English
```

Having done this, you have to set the correct path to the headers directory. You can do so by changing the following line. Remember that `../` means “go up to the parent folder”.

```
1 \def\headersdir{../../headers/}
```

You should also enter some information for the title and page headers. These include the name of the lecture (`\lecture`), the task sheet number (`\setsheetno`, remove to hide the sheet number), the due date (`\date`, remove to hide due date), and the authors (`\author`). Make sure to put these in the preamble (the code before `\begin{document}`).

```
1 \lecture{Foundation of \LaTeX}
2 \setsheetno{1}
3 \date{March 13}
4 \author{Dr. Herrmann Einstein (4201337)}
```

With all of this set, you are ready to write down your solutions. Have fun!

2 Task 2 – Special Task Sheet Macros

To mark the beginning of a new task, subtask, or sub-subtask use the following macros. You can specify a title for the given ((sub-)sub-)task by putting it in square brackets after the respective macro.

```

1 \ex
2 \subex
3 \subsubex
4
5 \ex[An awesome title]
6 \subex[An even more awesome title]
7 \subsubex[The most awesome title]
```

To change the style of numbering for the sheet or for tasks, you can use the following macros. These macros can also be used for ((sub-)sub-)tasks by replacing the \sheet part inside the macros by \ex, \subex, or \subsubex, respectively.

```

1 \sheetstylearabic % Arabic numerals (1, 2, 3, ...)
2 \sheetstyleroman % Lowercase roman numerals (i, ii, iii, ...)
3 \sheetstyleRoman % Uppercase roman numerals (I, II, III, ...)
4 \sheetstylealph % Lowercase latin alphabet (a, b, c, d)
5 \sheetstyleAlph % Uppercase latin alphabet (A, B, C, D)
```

To change the number of a sheet or task, you can use the following macros. Independent of the numbering style, the new number has to be provided in arabic numerals.

```

1 \setsheetno{1}
2 \setexno{3}
3 \setsubexno{3}
4 \setsubsubexno{7}
```

3 Task 3 – Typesetting Math

To make typesetting math more convenient, this template provides a plethora of macros that can be used in math mode.

Source Code	Result	Description
<code>\dif x</code>	dx	d for integrals and differentials
<code>\im x</code>	$\operatorname{Im} x$	Imaginary part
<code>\re x</code>	$\operatorname{Re} x$	Real part
45 <code>\dgr</code>	45°	Degree sign
<code>x \inv</code>	x^{-1}	Inverse
<code>x \tx {3}</code>	$x \cdot 10^3$	Power of ten (scientific notation)
<code>\abs {x}</code>	$ x $	Absolute value
<code>\brk {x}</code>	(x)	Round brackets
<code>\sbr {x}</code>	$[x]$	Square brackets
<code>\ceil {x}</code>	$\lceil x \rceil$	Euclidian round-up brackets
<code>\floor {x}</code>	$\lfloor x \rfloor$	Euclidian round-down brackets
<code>\set {x}</code>	$\{x\}$	Set
<code>\set {x \where y}</code>	$\{x \mid y\}$	Set with condition
<code>x \bdiv y</code>	$x \operatorname{div} y$	Integer division
<code>x \bmod y</code>	$x \operatorname{mod} y$	Integer division remainder
<code>\vcol</code>	$:$	Vertically centered colon
<code>\defeq</code>	$:=$	Definition operator
<code>\eqdef</code>	\equiv	Mirrored definition operator
<code>\NN</code>	\mathbb{N}	Set of Natural Numbers
<code>\ZZ</code>	\mathbb{Z}	Set of Integers
<code>\QQ</code>	\mathbb{Q}	Set of Rational Numbers
<code>\RR</code>	\mathbb{R}	Set of Real Numbers
<code>\CC</code>	\mathbb{C}	Set of Complex Numbers
<code>\PP</code>	\mathbb{P}	Set of Prime Numbers
<code>\HH</code>	\mathbb{H}	Set of Quarternions
<code>\BB</code>	\mathbb{B}	Set of Booleans
<code>\EE</code>	\mathbb{E}	Expected value
<code>\Ave</code>	Ave	Average
<code>\Var</code>	Var	Variance
<code>\Cov</code>	Cov	Covariance
<code>\Pr</code>	Pr	Probability
<code>\frac {x}{y}</code>	$\frac{x}{y}$	Fraction
<code>\sfrac {x}{y}</code>	x/y	Diagonal fraction
<code>\bsfrac {x}{y}</code>	y^x	Backwards diagonal fraction

Continued on the following page...

Source Code	Result	Description
<code>\bigO</code>	O	Landau symbols (Big-O notation)
<code>\R</code>	R	Recursive languages
<code>\RE</code>	RE	Recursively enumerable languages
<code>\REG</code>	REG	Regular languages
<code>\coRE</code>	$co-RE$	Complement-recursively enumerable languages
<code>\REC</code>	REC	Recursively decidable languages
<code>\P</code>	P	Polynomial-time problems
<code>\NP</code>	NP	Nondeterministic poly-time problems
<code>\coNP</code>	$co-NP$	Complement-nondeterministic poly-time problems
<code>\DSpace</code>	$DSpace$	Deterministic space class
<code>\NSpace</code>	$NSpace$	Nondeterministic space class
<code>\DTime</code>	$DTime$	Deterministic time class
<code>\NTime</code>	$NTime$	Nondeterministic time class
<code>\emptyword</code>	ε	Empty word
<code>\emptyset</code>	\emptyset	Empty set
<code>\elem</code>	\in	Element of
<code>\notelem</code>	\notin	Not Element of

Apart from these new macros, the template loads the `amsmath` and `amsthm` packages to provide the most relevant math macros and environments.

4 Task 4 – Displaying Code

4.1 Task 4.a – Real code

To render code into your submissions, you can use the features provided by minted. Take the following python snippet.

```

1 def fib(n):
2     if n == 0 or n == 1:
3         return 1
4     return fib(n - 1) + fib(n - 2)

```

You can render it using this code:

```

1 \begin{minted}{python}
2     def fib(n):
3         if n == 0 or n == 1:
4             return 1
5         return fib(n - 1) + fib(n - 2)
6 \end{minted}

```

4.2 Task 4.b – Pseudocode

To produce pseudocode, you can use the pseudo environment.

Take this sample program from the TCS lecture:

Program 1: Construction P	
1	P' ;
2	if $x_1 \neq 0$ then
3	$x_0 := x_0$
4	else
5	for x_0 do
6	Do nothing...
7	od
8	fi
9	;
10	if $x_0 = 0$ then
11	$x_0 := 1$;
12	while $x_0 \neq 0$ do
13	$x_1 := 1$
14	od
15	fi

You can render it using this code:

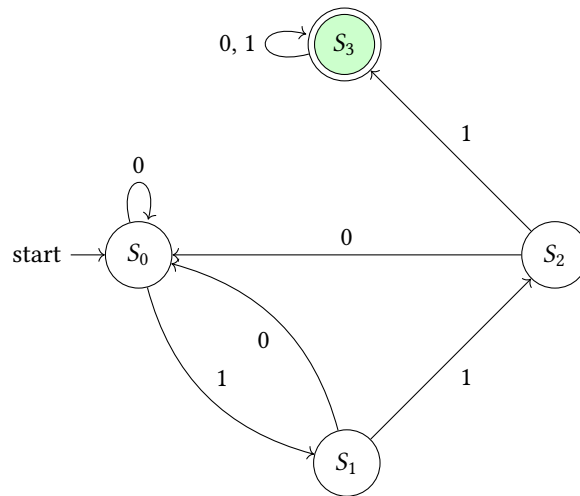
```

\begin{pseudo}
    $P'$\;
    \eIf{$x_1 \neq 0$}{
        $x_0 \defeq x_0$
    }{
        \For{$x_0$}{
            Do nothing...
        }
    };
    \If{$x_0 = 0$}{
        $x_0 \defeq 1$;
        \While{$x_0 \neq 0$}{
            $x_1 \defeq 1$
        }
    }
    \caption{Construction $P$}
\end{pseudo}

```

5 Task 5 – Drawing Automata

You can use TikZ to draw automata. See the following example:



You can produce it using this code:

```

1 \begin{tikzpicture}[node distance=3cm, auto]
2   % Define states
3   \node[state, initial] (s0) {$S_0$};
4   \node[state, below right = of s0] (s1) {$S_1$};
5   \node[state, above right = of s1] (s2) {$S_2$};
6   \node[state, accepting, above left = of s2] (s3) {$S_3$};
7
8   % Define transitions
9   \path[->]
10    (s0) edge [loop above] node {0} ()
11    (s3) edge [loop left] node {0, 1} ()
12
13    (s0) edge [bend right] node {1} (s1)
14    (s1) edge [bend right] node {0} (s0)
15    (s1) edge [swap] node {1} (s2)
16    (s2) edge [swap] node {1} (s3)
17    (s2) edge [swap] node {0} (s0)
18 ;
19 \end{tikzpicture}

```
