Wie kann ich die Möglichkeiten von Python für LaTeX nutzen?

Zusammenspiel von Python und LaTeX

Günter Partosch

Justus-Liebig-Universität Gießen, Hochschulrechenzentrum (HRZ)

Version 1.0.2, 19. Oktober 2019

Inhalt des Vortrags

Zusammenspiel von Python und LaTeX

Python

PythonTeX

Beispiele

Literatur

Zusammenfassung

Python hat sich in den letzten Jahren in Schule und Hochschule zu einer der wichtigsten Programmiersprache entwickelt. Das ist einmal begründet in der einfachen Syntax als auch in der großen Flexibilität. Zahlreiche Python-Module erweitern den Funktionsumfang und ermöglichen interessante Anwendungen. Mit ihr lassen sich

- Gleichungssysteme und aufwändige nummerische und statistische Aufgaben lösen,
- Daten visualisieren,
- Ableitungen und Integrale symbolisch darstellen,
- Werte tabellieren,
- ...

Im Vortrag werden anhand mehrerer Beispiele zwei Wege vorgestellt, wie diese Fähigkeiten in LaTeX genutzt werden können.

Links
Die jeweils neueste Version des Vortrags finden Sie unter http://www.staff.uni-giessen.de/partosch/unterlagen/python+latex-alles.pdf, die Beispiele unter http://www.staff.uni-giessen.de/partosch/unterlagen/python-latex-beispiele.zip.

Zusammenspiel von Python und LaTeX

Zusammenspiel von Python und LaTeX

Python

PythonTeX

Beispiele

Literatur

Wie kann ich die Vorteile von Python in/für LaTeX nutzen?

Zusammenspiel von Python und LaTeX (1)

Python kann

- Gleichungssysteme und aufwändige nummerische und statistische Aufgaben lösen,
- Daten visualisieren,
- Ableitungen und Integrale symbolisch darstellen,
- Werte tabellieren, ...

Python kann aber *nicht*

- die Ergebnisse als hochwertige PDF-Dokumente ausgeben.

LaTeX kann

- qualitativ hochwertige PDF-Dokumente erstellen

LaTeX verfügt aber *nicht*

- über die oben genannten Fähigkeiten von Python.

Wie kann ich die Vorteile von Python in/für LaTeX nutzen?

Zusammenspiel von Python und LaTeX (2): zwei Möglichkeiten für die Zusammenarbeit

Python als Ausgangspunkt:

- alle Möglichkeiten von Python *direkt* nutzbar
- hochwertige PDF-Dokumente nicht direkt erstellbar
- Ausweg: mittels `print` Zeilen eines LaTeX-Dokuments erzeugen → übersetzen → PDF-Dokument
- LaTeX-Dokument muss vollständig und korrekt sein.
- möglicher Workflow:
  `python` *python-programm* > *latex-datei*
  x`latex` *latex-datei*

LaTeX als Ausgangspunkt:

- LaTeX generiert den Ausgabetext.
- Python wird *indirekt* (über das LaTeX-Paket `pythontex`) genutzt: Python-Code in spezielle Anweisungen einbetten → an Python übergeben → übersetzen → ausführen → betreffende Ausgabe ggf. ins LaTeX-Dokument einfügen
- zusätzliche Möglichkeiten zur Manipulation von Code und Ausgabe

Python

PythonTeX
G. Partosch
Py+LaTeX
Python
Typen
Strings
Keys
Kontrolle
Module
pythonTeX
Beispiele
Literatur

Zusammenspiel von Python und LaTeX

Python
- Datentypen
- Zeichenketten
- Schlüsselwörter
- Kontrollstrukturen
- Module

PythonTeX

Beispiele

Literatur

Python

PythonTeX
G. Partosch
Py+LaTeX
Python
Typen
Strings
Keys
Kontrolle
Module
pythonTeX
Beispiele
Literatur

- entwickelt ab 1991 von Guido van Rossum; später von der *Python Software Foundation*; aktuelle Version 3.7.4 (8.7.2019)
- ein **Entwicklungsziel**: gut lesbarer, knapper Programmierstil $\Longrightarrow$ Unterrichtssprache an Schulen und Hochschulen
  - untergeordnete Blöcke werden eingerückt $\rightarrow$ nicht geklammert
  - ganz zeilenorientiert (mit wenigen Ausnahmen) $\rightarrow$ 1 Anweisung pro Zeile
  - nur noch wenige Schlüsselwörter
- Python unterstützt u. a. **objektorientierte, strukturierte und funktionale Programmierung**
- **dynamische Datentypisierung**: Wert bestimmt den jeweiligen Typ; keine statische Typüberprüfung
- gut ausgebaute und gut gepflegte **Standardbibliothek** als Basis [van Rossum et al. 2019, `reference.pdf`]; Erweiterungen über zahlreiche Module [van Rossum et al. 2019, `library.pdf`]
- **Tutorium** [van Rossum et al. 2019, `tutorial.pdf`]
- **FAQs** [van Rossum et al. 2019, `faq.pdf`]
- **HowTos** zu verschiedenen Themen, z. B. Sortierung, reguläre Ausdrücke, funktionale Programmierung

Python – Datentypen (1)

einfache Typen (1)

- ganzzahlige Werte (`int`)
- Fließkomma-Werte (`float`)
- logische Werte (`bool`)
- komplexe Werte (`complex`)

Ausflug: Zuweisungen

- `a = 23` `# ganzzahliger Wert`
- `a = 1.2; b = 2.34e-20` `# Fließkomma-Werte; 2 Anw. in Zeile`
- `a, b = 1, 20` `# bedeutet: a = 1; b = 20`
- `a = b = 5` `# bedeutet: b = 5; a = b; Mehrfachzuweisung`
- `a += 25` `# a = a + 25`
- `c1 = 2.3 + 4.5j` `# komplexe Zahl`

zulässige Operatoren

Typ	<	>	==	!=	>=	<=	+	-	*	**	/	//	%
ganzzahlig	x	x	x	x	x	x	x	x	x	x	x	x	x
Fließkomma	x	x	x	x	x	x	x	x	x	x	x		
logisch			x	x									
komplex			x	x			x	x	x	x	x		

Operatoren bei `bool`: `and`, `or`, `not`

Python – Datentypen (2)

einfache Typen (2)

einige Funktionen für diese Datentypen

Funktion	Bedeutung
`abs(`*zahl*`)`	Absolutbetrag von *zahl*
`bin(`*zahl*`)`	binäre Darstellung von *zahl*
`divmod(`*zahl1*`,` *zahl2*`)`	liefert Tupel (*zahl1* // *zahl2*; *zahl1* % *zahl2*)
`float(`*zahl*`)`	Gleitpunkt-Darstellung von *zahl*
`hex(`*zahl*`)`	hexadezimale Darstellung von *zahl*
`int(`*zahl*`)`	ganzzahlige Darstellung von *zahl*
`oct(`*zahl*`)`	oktale Darstellung von *zahl*
`chr(`*zahl*`)`	liefert das Zeichen mit der Codierung *zahl*
`str(`*ausdruck*`)`	liefert zugehörige Zeichenketten-Repräsentierung

Python – Datentypen (3)

zusammengesetzte Typen (1)

Listen

```
werte = [2, "a", 1.3]  # einzelne Elemente einer Liste durch Aufzählung
werte.append("abc")    # neues Element am Ende
werte.insert(1, 3.4)   # neues Element an der Position 1 (2. Element!)
```

Tupel

```
werte = (2, "a", 1.3)  # einzelne Elemente durch Aufzählung
#werte.append("abc")   # Anhängen nicht möglich
#werte.insert(1, 3.4)  # Einfügen nicht möglich
```

einige gemeinsame Konstrukte

```
l1  = [1, 2]
l2  = [3, 4]
l3  = l1 + l2            # Verkettung zweier Listen
t1  = (2, 3) + (3, 4, 5)# Verkettung zweier Tupel
i1  = len(l3)            # Anzahl der Elemente einer Liste
l10 = t1[0]              # 1. Element
i2  = l3[i1 - 1]         # letztes Element
l4  = l3[1:4]            # 2. bis 4. Element
```

Python – Datentypen (4)

zusammengesetzte Typen (2)

und dann gibt es noch Mengen (set)

```
s1 = set('abc')  # Menge mit den Elementen a, b, c
s2 = set('bcd')  # Menge mit den Elementen b, c, d
print(s1 - s2)   # Differenzmenge: {'a'}
print(s1 | s2)   # Vereinigungsmenge: {'b', 'c', 'd', 'a'}
print(s1 & s2)   # Schnittmenge: {'b', 'c'}
print(s1 ^ s2)   # Symmetrische Differenz: {'d', 'a'}
print('a' in s1) # Element enthalten in s1: True
print(len(s1))   # Mächtigkeit einer Menge: 3
```

und Wörterbücher (dictionary)

```
staedte ={'Gießen' : 75000, 'Marburg' : 60000}
for s in staedte:
    print(s, "mit", staedte[s], "Einwohner")
```

Python – Datentypen (5)

Zeichenketten

- Zeichenkette (`string`) in Python: Folge von Zeichen
- nach ihrer Vereinbarung ist eine Python-Zeichenkette unveränderlich

Zeichenketten-Darstellungen

```
z1 = 'eine Zeichenkette mit Gänsefüßchen (") aber ohne Apostroph\n'
z2 = "eine Zeichenkette ohne Gänsefüßchen aber mit Apostroph (')"
z3 = """mit eingebetteten
      Zeilenenden und Anführungszeichen (' ")"""
z4 = r"\section{Eine Überschrift in \LaTeX}" + "\n"        # raw string
```

einige String-Methoden/Funktionen

Funktion/Methode	Bedeutung
string`.capitalize()`	Zeichenkette mit gr. Anfangsbuchstaben
string`.upper()`	Zeichenkette nur aus Großbuchstaben
string`.lower()`	Zeichenkette nur aus Klienbuchstaben
`len(`*string*`)`	Länge von *string*
string`.find(`*string1*`)`	findet Position von *string1* in *string*
string`.replace(`*string1*`, `*string2*`)`	ersetzt *string1* in *string* durch *string2*
string`.split(`*trenner*`)`	trennt *string* bei *trenner* auf; liefert Liste
string`.isdigit()`	`True`, falls *string* nur Ziffern enthält

Python – Schlüsselwörter

PythonTeX
G. Partosch
Py+LaTeX
Python
Typen
Strings
Keys
Kontrolle
Module
pythonTeX
Beispiele
Literatur

and	def	finally	in	or	while
as	del	for	is	pass	with
assert	elif	from	lambda	raise	yield
break	else	global	None	return	
class	except	if	nonlocal	True	
continue	False	import	not	try	

Schlüsselwörter sind Bestandteile der Sprache Python
$\Longrightarrow$ keine eigene Bezeichner

Python – Kontrollstrukturen (1)

`if`-Anweisung

```
if a <= 2:
    print("Der Wert ist nicht größer als 2.")
```

`if`-Anweisung mit einer Alternative

```
if a <= 2:
    print("Der Wert ist nicht größer als 2.")
else:
    print("Der Wert ist größer als 2.")
```

`if`-Anweisung mit mehreren Alternativen

```
if auswahl in ["Ja", "j"]:
    print("Ja")
elif auswahl in ["Nein", "n"]:
    print("Nein")
else:
    print("keine richtige Antwort")
```

Python – Kontrollstrukturen (2)

for-Anweisung

```
for t in ["Günter Partosch", "Emil Mayer"]:
    print('Der Teilnehmer heißt:', t)
```

for-Anweisung mit range()

```
summe = 0                                   # Initialisierung
for i in range(1, 21):                      # Summe der Zahlen 1 bis 20
    summe = summe + i
```

while-Anweisung

```
summe = 0; i = 1                            # Initialisierung
while i <= 20:
    summe = summe + i; i += 1
```

Python – Module

PythonTeX
G. Partosch
Py+LaTeX
Python
Typen
Strings
Keys
Kontrolle
Module
pythonTeX
Beispiele
Literatur

- erweitern den Funktionsumfang von Python
- oder nehmen Einstellungen vor
- Übersicht und Beschreibungen in [van Rossum et al. 2019, `library.pdf`]

Import von Modulen

`import` *quelle*
`import` *quelle* `as` *name*
`from` *quelle* `import` [`*` | *name(n)*]

kleine Auswahl

`csv`	CSV-Daten verarbeiten	`randassign`	Zufallszahlen
`math`	mathematische Funktionen	`random`	Zufallszahlen
`matplotlib`	2D-Grafiken erstellen	`re`	reguläre Ausdrücke
`numpy`	Numerische Mathematik	`subprocess`	Sub-Prozesse
`pickle`	Python-Objekte serialisieren	`sympy`	symbolische Mathematik
`pyx`	PS/PDF-Grafiken erstellen	`time`	Zeitmessungen

PythonTEX

PythonTeX
G. Partosch
Py+LaTeX
Python
pythonTeX
Installation
Parameter
Workflow
Anweisung
Beispiele
Literatur

Zusammenspiel von Python und LATEX

Python

PythonTEX
- Installation
- Parameter, Optionen
- Workflow
- Anweisungen

Beispiele

Literatur

PythonTEX
Entwicklungsprinzipien und Allgemeines

PythonTeX
G. Partosch
Py+LaTeX
Python
pythonTeX
Installation
Parameter
Workflow
Anweisung
Beispiele
Literatur

- PythonTEX [Poore 2019];
  einführende Texte in [Poore 2015, Mertz et al. 2013, Gosling 2016]
  aktuelle Version: 0.17
- Python-Code in ein LATEX-Dokument einbetten
  - an Python übergeben → übersetzen → ausführen → betreffende Ausgabe ggf. in LATEX-Dokument einfügen
  - Python-Code wird nur dann erneut übersetzt, wenn er geändert wurde
  - Python-Code ggf. in Sessions aufteilbar, die parallel abgearbeitet werden
- Python-Code kann aufbereitet im LATEX-Dokument ausgegeben werden (prettyprinting)
- PythonTEX unterstützt derzeit direkt die Python-Module `pylab` (aus `matplotlib` und `numpy`) und `sympy`

PythonTEX – Installation

PythonTeX
G. Partosch
Py+LaTeX
Python
pythonTeX
Installation
Parameter
Workflow
Anweisung
Beispiele
Literatur

Anforderungen/Voraussetzungen

- aktuelle TEX/LATEX-Installation (MiKTEX/TEXLive)
- oder Download von https://github.com/gpoore/pythontex
- LATEX-Pakete: `fancyvrb`, `fvextra`, `etoolbox`, `xstring`, `pgfopts`, `newfloat`, `currfile`, `color/xcolor`
- ggf. `graphicx` – `mdframed` oder `tcolorbox` oder `framed`
- aktuelle Python-Installation (aktuell 3.7.4)
- ggf. Python-Module: `pygments` – `numpy`, `scipy`, `matplotlib`, `sympy`
- Editoren/Entwicklungsumgebungen für LATEX und Python

Dateien nach der Installation (unter Windows)

- `pythontex.sty` in `C:/texlive/2019/texmf-dist/tex/latex/pythontex`
- `syncpdb.py` in `C:/texlive/2019/texmf-dist/doc/latex/pythontex`
- bei mir sonst in `C:/texlive/2019/texmf-dist/scripts/pythontex`
- `pythontex.py`
- `pythontex_engines.py`
- `pythontex_utils.py`
- `depythontex.py`
- `pythontex_install.py`

PythonTEX – Optionen des Pakets pythontex

Optionen des LaTeX-Pakets pythontex (Auswahl)	[Poore 2019, 4.1]
debug	Debugging ermöglichen
depythontex= true | false	zusätzliche Datei für den Aufruf von depythontex erzeugen
hashdependencies= true | false	auf geänderte externe Dateien überprüfen
prettyprinter= pygments | fancyvrb	Paket/Programm für Prettyprinting
prettyprintinline= true | false	Inline-Prettyprinting erlauben
pyginline= true | false	eingebettete Pygments-Ausgabe erlauben
pygments= true | false	Pygments-Ausgabe erlauben
rerun= never | modified | errors | warnings | always	bei bestimmten Bedingungen erneut übersetzen
runall= true | false	auch wenn Python-Anteile nicht geändert wurden, erneut übersetzen
upquote= true | false	aufrechte Anführungszeichen

Üblicherweise werden die Optionen als Parameter an das Programm pythontex weiter geleitet.

PythonTEX – Parameter des Programms pythontex

Aufruf des Programms pythontex

pythontex [*parameter*] *datei*[`.pytxcode`]

Optionale Parameter des Programms pythontex (Auswahl) [Poore 2019, 3.2]

`--error-exit-code true` | `false`	liefert auch bei Fehler den Fehlercode 1
`--runall` [`true` | `false`]	wie die Paket-Option `runall`
`--rerun never` | `modified` | `errors` | `warnings` | `always`	wie die Paket-Option `rerun`
`--hashdependencies` [`true` | `false`]	wie die Paket-Option `hashdependencies`
`--jobs` n	erlaubt n gleichzeitige Jobs (Voreinstellung: `cpu_count()`)
`--verbose`	»geschwätzige« Ausgabe

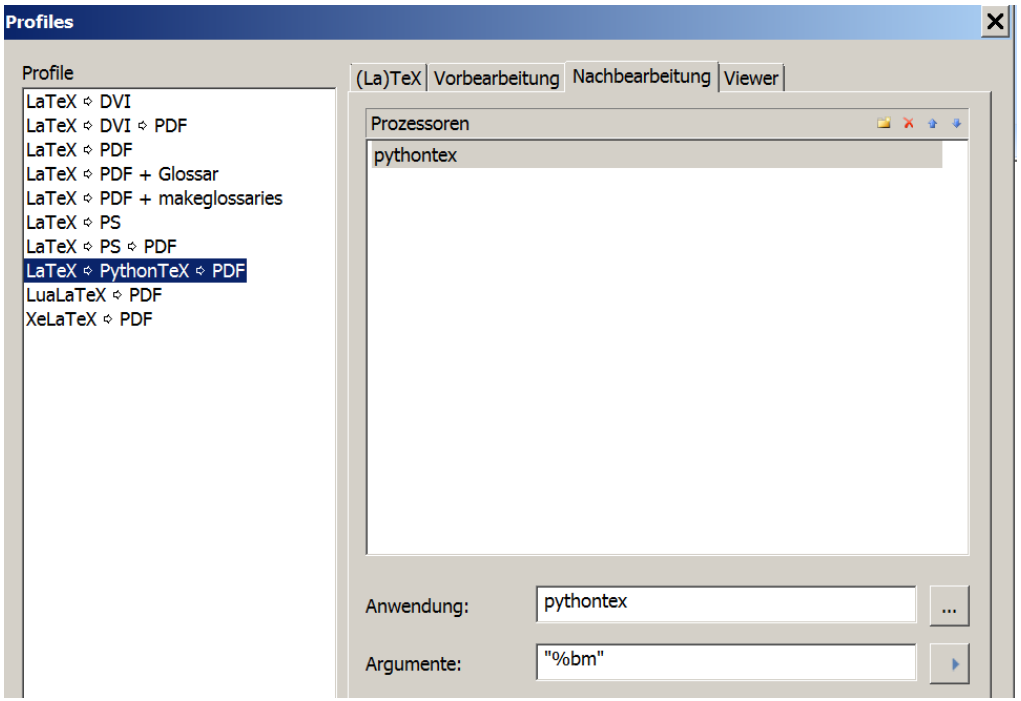
PythonTEX – Workflow (1)

im einfachsten Fall auf der Kommandozeile

PythonTEX – Workflow (1)

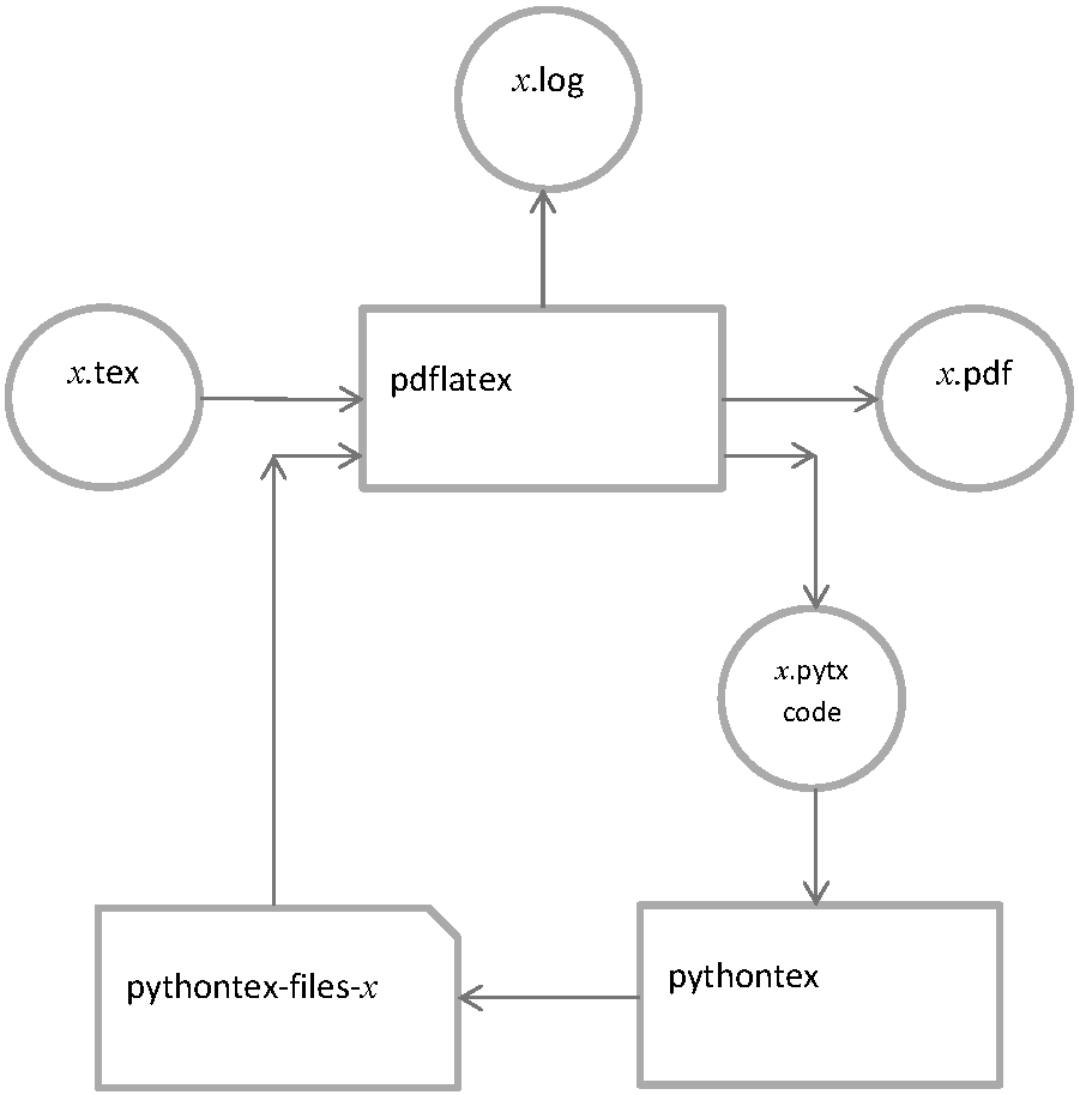
*x*latex *datei*[.tex]
pythontex [*parameter*] *datei*[.pytxcode]
*x*latex *datei*[.tex]
...

beim Einsatz einer LATEX-Benutzeroberfläche (beispielsweise TEXnicCenter)



PythonTEX – Workflow (2)

PythonTeX
G. Partosch
Py+LaTeX
Python
pythonTeX
Installation
Parameter
Workflow
Anweisung
Beispiele
Literatur



PythonTEX – Anweisungen

\py und verwandte Anweisungen und Umgebungen (1)

\py und verwandte Anweisungen und Umgebungen (1) [Poore 2019]

Python		Python + Modul pylab		
Anweisung	Umgebung	Anweisung	Umgebung	
\py{*ausdruck*}		\pylab{*ausdruck*}		W
\pyc{*code*}	pycode	\pylabc{*code*}	pylabcode	C + A
\pyb{*code*}	pyblock	\pylabb{*code*}	pylabblock	C + P
\pyv{*code*}	pyverbatim	\pylabv{*code*}	pylabverbatim	P

\py und verwandte Anweisungen und Umgebungen (2) [Poore 2019]

Python + Modul sympy		
Anweisung	Umgebung	
\sympylab{*ausdruck*}		W
\sympyc{*code*}	sympycode	C + A
\sympyb{*code*}	sympyblock	C + P
\sympyv{*code*}	sympyverbatim	P

W: Ausgabe des Werts von *ausdruck*
C + A: Ausführung von *code* und lediglich Ausgabe auf die Standardausgabe
C + P: Ausführung von *code* und Prettyprinting mittels \printpythontex
P: nur Prettyprinting

Beispiele

Zusammenspiel von Python und LaTeX

Python

PythonTeX

Beispiele

Literatur

Beispiele

Gerüste für die folgenden Beispiele (1)

Gerüst für Python-Beispiele

```
print(r"\documentclass[paper=a4, fontsize=11, parskip=half]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{graphicx}")

print(r"\begin{document}")
print(r"\section*{...}")

# Platz für weitere Python-Anweisungen, die ggf.
# weiteren LaTeX-Code erzeugen
#

print(r"\end{document}")
```

Beispiele

Gerüste für die folgenden Beispiele (2)

Gerüst für PythonTeX-Beispiele

```
\documentclass[paper=a4, fontsize=11, parskip=half]{scrartcl}
\usepackage{fontspec}
\usepackage{pythontex}
\usepackage{graphicx}

\begin{document}
\section*{...}

% Text + weitere LaTeX-Anweisungen

\begin{pycode}
# Python-Code
\end{pycode}

% Text
% + weitere LaTeX-Anweisungen
% + ggf. PythonTeX-Code
%

\end{document}
```

Tabelle (2^m)

gesucht

- Tabelle mit 2^m, $m = 1, \ldots, 40$

Lösung

- `for`-Schleife

Python-Datei `python5.py` $\Longrightarrow$ LaTeX-Datei `tex5.tex`
PythonTeX-Datei `pytex5.tex` $\Longrightarrow$ PDF-Datei `pytex5.pdf`

Tabelle ($\sum_{i=1}^{40} i$, F_m, $m!$)

gesucht

- Tabelle mit $\sum_{i=1}^{40} i$
- Tabelle mit $F_m, m = 1, \ldots, 40$
- Tabelle mit $m!, m = 1, \ldots, 25$

Lösung

- Python-Modul `math` mit der Methode `factorial`
- `for`-Schleifen
- eigene Funktionen `tabelliereFunktion(...)`, `fibo(n)`, `summe(n)`

Python-Datei `python8.py` $\Longrightarrow$ LaTeX-Datei `tex8.tex`
PythonTeX-Datei `pytex8.tex` $\Longrightarrow$ PDF-Datei `pytex8.pdf`

PythonTeX
G. Partosch
Py+LaTeX
Python
pythonTeX
Beispiele
Literatur

Tabellen $(2,3,4,5)^m$, $m * n$, $m + n$, m^n

gesucht

- Tabelle für $(2,3,4,5)^m$, $m = 0, \ldots, 30$
- Tabelle für $m * n$; $m, n = 0, \ldots, 9$
- Tabelle für $m + n$; $m, n = 0, \ldots, 9$
- Tabelle für m^n; $m, n = 0, \ldots, 9$

Lösung

- geschachtelte `for`-Schleifen

Python-Datei `python20.py` $\Longrightarrow$ LaTeX-Datei `tex20.tex`
PythonTeX-Datei `pytex20.tex` $\Longrightarrow$ PDF-Datei `pytex20.pdf`

graphische Darstellung ($y = \cos(2\pi t)e^{-t}$)

$$y = \cos(2\pi t)e^{-t}$$

gesucht

- Graph der Funktion (gedämpfte Schwingung)

Lösung

- Python-Modul `pylab` (graphische Darstellung)

Python-Datei `python38.py` $\Longrightarrow$ LaTeX-Datei `tex38.tex`
PythonTeX-Datei `pytex38.tex` $\Longrightarrow$ PDF-Datei `pytex38.pdf`

PythonTeX
G. Partosch
Py+LaTeX
Python
pythonTeX
Beispiele
Literatur

graphische Darstellung ($y(x) = \frac{\sin(x)}{x}; y(x) = \frac{\cos(x)}{x}$)

$$y(x) = \frac{\sin(x)}{x}$$
$$y(x) = \frac{\cos(x)}{x}$$

gesucht

- Graphen der Funktionen

Lösung

- Python-Modul `pyx` (graphische Darstellung)

Python-Datei `python22.py` $\Longrightarrow$ LaTeX-Datei `tex22.tex`
PythonTeX-Datei `pytex22.tex` $\Longrightarrow$ PDF-Datei `pytex22.pdf`

Primzahlen

gesucht

- die ersten 500 Primzahlen

Lösung

- Python-Modul `sympy`
- `sympy.prime(`*integer*`)`
- `for`-Schleife
- eigene Python-Funktion `Primzahlen(n)`

Python-Datei `python11.py` $\Longrightarrow$ LaTeX-Datei `tex11.tex`
PythonTeX-Datei `pytex11.tex` $\Longrightarrow$ PDF-Datei `pytex11.pdf`

Binomial-Ausdrücke

gesucht

- $(a+b)^n, n = 0, \ldots, 10$

Lösung

- Python-Modul `sympy`
- `sympy.var(`*zeichenkette*`)`
- `sympy.latex(`*sympy-ausdruck*`)`
- `sympy.expand(`*sympy-ausdruck*`)`
- `for`-Schleifen

Python-Datei `python12.py` $\Longrightarrow$ LaTeX-Datei `tex12.tex`
PythonTeX-Datei `pytex12.tex` $\Longrightarrow$ PDF-Datei `pytex12.pdf`

lineares Gleichungssystem

gesucht

- Ausgabe von Gleichungssystem und Lösung

Lösung

- Python-Modul `sympy`
- `sympy.var(`*zeichenkette*`)`
- `sympy.solve(`*gls*`)`
- `sympy.latex(`*sympy-ausdruck*`)`
- `for`-Schleifen

Python-Datei `python24a.py` $\Longrightarrow$ LaTeX-Datei `tex24a.tex`
PythonTeX-Datei `pytex24a.tex` $\Longrightarrow$ PDF-Datei `pytex24a.pdf`

Statistik (Korrelationskoeffizienten)

gesucht

- Auflistung von Versuchsdaten
- Berechnung des Korrelationskoeffizienten (nach Pearson bzw. Spearman)

Lösung

- Python-Modul `scipy.stats` (Statistik)
- zeilenweises Einlesen der Daten
- Berechnung des Korrelationskoeffizienten mittels der Methoden `scipy.stats.spearmanr` und `scipy.stats.pearsonr`
- zeilenweise Ausgabe der Daten
- `for`-Schleifen

Python-Datei `python52.py` $\Longrightarrow$ LaTeX-Datei `tex52.tex`
PythonTeX-Datei `pytex52.tex` $\Longrightarrow$ PDF-Datei `pytex52.pdf`

Statistik und graphische Darstellung (Streudiagramm, Regressionsgerade)

gesucht

- Darstellung von Versuchsdaten in einem Streudiagramm
- Berechnung und Darstellung der Regressionsgeraden

Lösung

- Python-Modul `matplotlib.pyplot` (graphische Darstellung)
- Python-Modul `pandas` (Einlesen und Aufbereiten der Daten)
- Python-Modul `sklearn.linear_model` (Berechnung der Regressionsgeraden)
- Methode `pandas.read_csv`

Python-Datei `python51.py` $\Longrightarrow$ LaTeX-Datei `tex51.tex`
PythonTeX-Datei `pytex51.tex` $\Longrightarrow$ PDF-Datei `pytex51.pdf`

unbestimmte Integrale

$$\int \sin(x)\,dx;\ \int \cos(x)\,dx;\ \int \tan(x)\,dx;\ \int \cot(x)\,dx$$

gesucht

- formale Berechnung der Integrale

Lösung

- Python-Modul `sympy`
- `sympy.var(`*zeichenkette*`)`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`)`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`).doit()`
- `sympy.latex(`*sympy-ausdruck*`)`
- `for`-Schleife
- in `sympy` bekannt: `sin(x)`, `cos(x)`, `tan(x)`, `cot(x)`

Python-Datei `python13.py` $\Longrightarrow$ LaTeX-Datei `tex13.tex`
PythonTeX-Datei `pytex13.tex` $\Longrightarrow$ PDF-Datei `pytex13.pdf`

unbestimmtes und bestimmtes Integral

$$f(x) = \int x^3 + \cos^5(x)\,dx \qquad g(\phi) = \int_0^\infty e^{-\phi^2}\,d\phi$$

gesucht

- formale Berechnung der Integrale und Bestimmung des nummerischen Werts

Lösung

- Python-Modul `sympy`
- `sympy.var(`*zeichenkette*`)`, `sympy.Symbol(`*zeichenkette*`)`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`)`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`,` *variable*`).doit()`
- `sympy.latex(`*sympy-ausdruck*`)`
- `sympy.N(`*sympy-ausdruck*`)`
- in `sympy` bekannt: `cos(x)`, `oo`, `exp(x)`, `pi`

Python-Datei `python14.py` $\Longrightarrow$ LaTeX-Datei `tex14.tex`
PythonTeX-Datei `pytex14.tex` $\Longrightarrow$ PDF-Datei `pytex14.pdf`

bestimmtes 3-fach-Integral

$$f(x) = \int_0^4 \int_0^3 \int_0^2 xy + y\sin(z) + \cos(x+y)\, dx\, dy\, dz$$

gesucht

- schrittweise Darstellung der Berechnung

Lösung

- Python-Modul `sympy`
- `sympy.symbols(`*zeichenkette*`)`, `Symbol(`*zeichenkette*`)`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`)`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`).doit()`
- `sympy.latex(`*sympy-ausdruck*`)`
- `sympy.N(`*sympy-ausdruck*`)`
- in `sympy` bekannt: `sin(x)`, `cos(x)`

Python-Datei `python16.py` $\Longrightarrow$ LaTeX-Datei `tex16.tex`
PythonTeX-Datei `pytex16.tex` $\Longrightarrow$ PDF-Datei `pytex16.pdf`

PythonTeX
G. Partosch
Py+LaTeX
Python
pythonTeX
Beispiele
Literatur

Ableitungen ($\sin'(x)$, $\sinh'(x)$, $\csc'(x)$)

$$\frac{\mathrm{d}}{\mathrm{d}x}\sin(x) \qquad \frac{\mathrm{d}}{\mathrm{d}x}\sinh(x) \qquad \frac{\mathrm{d}}{\mathrm{d}x}\csc(x)$$

Lösung

- Python-Modul `sympy`
- `sympy.symbols(`*zeichenkette*`)`
- `sympy.Derivative(`*sympy-ausdruck*`,` *variable*`)`
- `sympy.Derivative(`*sympy-ausdruck*`,` *variable*`).doit()`
- `sympy.latex(`*sympy-ausdruck*`)`
- *zeichenkette*`.replace(...)`
- `for`-Schleife
- in `sympy` bekannt: `sin(x)`, `sinh(x)`, `csc(x)`

Python-Datei `python43.py` $\Longrightarrow$ LaTeX-Datei `tex43.tex`
PythonTeX-Datei `pytex43.tex` $\Longrightarrow$ PDF-Datei `pytex43.pdf`

Beispiele

Beispiel 15

Ableitungen und bestimmte Integrale

gesucht

- Tabelle mit Ableitungen und unbestimmten Integralen für 27 Funktionen

Lösung

- Python-Modul `sympy`
- `sympy.var(`*zeichenkette*`)`
- `sympy.Derivative(`*sympy-ausdruck*`,` *variable*`)`
- `sympy.Derivative(`*sympy-ausdruck*`,` *variable*`).doit()`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`)`
- `sympy.Integral(`*sympy-ausdruck*`,` *variable*`).doit()`
- `sympy.latex(`*sympy-ausdruck*`)`
- `eval(`*zeichenkette*`)`
- `for`-Schleife, `if`-Abfrage

Python-Datei `python23.py` $\Longrightarrow$ LaTeX-Datei `tex23.tex`
PythonTeX-Datei `pytex23.tex` $\Longrightarrow$ PDF-Datei `pytex23.pdf`

Literatur

Zusammenspiel von Python und LaTeX

Python

PythonTeX

Beispiele

Literatur

Literatur/Links (1)

Python

van Rossum, Guido; Python development team: *Functional Programming HOWTO – Release 3.7.4*; 2019; als `howto-functional.pdf` in https://docs.python.org/3/archives/python-3.7.4-docs-pdf-a4.zip; hrsg. von Python Software Foundation; zuletzt besucht am 2019-10-02

van Rossum, Guido; Python development team: *Python Frequently Asked Questions – Release 3.7.4*; 2019; als `faq.pdf` in https://docs.python.org/3/archives/python-3.7.4-docs-pdf-a4.zip; hrsg. von Python Software Foundation; zuletzt besucht am 2019-10-02

van Rossum, Guido; Python development team: *Python Tutorial – Release 3.7.4*; 2019; als `tutorial.pdf` in https://docs.python.org/3/archives/python-3.7.4-docs-pdf-a4.zip; hrsg. von Python Software Foundation; zuletzt besucht am 2019-10-02

van Rossum, Guido; Python development team: *Regular Expression HOWTO – Release 3.7.4*; 2019; als `howto-regex.pdf` in https://docs.python.org/3/archives/python-3.7.4-docs-pdf-a4.zip; hrsg. von Python Software Foundation; zuletzt besucht am 2019-10-02

van Rossum, Guido; Python development team: *Sorting HOW TO – Release 3.7.4*; 2019; als `howto-sorting.pdf` in https://docs.python.org/3/archives/python-3.7.4-docs-pdf-a4.zip; hrsg. von Python Software Foundation; zuletzt besucht am 2019-10-02

Literatur/Links (2)

van Rossum, Guido; Python development team: *The Python Language Reference – Release 3.7.4*; 2019; als `reference.pdf` in https://docs.python.org/3/archives/python-3.7.4-docs-pdf-a4.zip; hrsg. von Python Software Foundation; zuletzt besucht am 2019-10-02

van Rossum, Guido; Python development team: *The Python Library Reference – Release 3.7.4*; 2019; als `library.pdf` in https://docs.python.org/3/archives/python-3.7.4-docs-pdf-a4.zip; hrsg. von Python Software Foundation; zuletzt besucht am 2019-10-02

PythonTeX

Abbasi, Nasser M.: *A very simple introduction to using Python in Latex*;; 2015;; http://www.12000.org/my_notes/python_in_latex/index.pdf;; zuletzt besucht am 2019-10-02

Dautermann, Wolfgang: *Programmierung mit LaTeX – . . . und anderen Programmiersprachen*; 2014; hrsg. von FH Johanneum; http://wolfgang.dautermann.at/vortraege/Linuxday-2014-Programmierung-mit-Latex.pdf; zuletzt besucht am 2019-10-02

Giacomelli, Roberto; Pignalberi, Gianluca: *Typesetting and highlighting Unicode source code with LaTeX – a package comparison*; 2014; in *Ars TeXnica* (18), S. 39–44; http://www.guitex.org/home/images/ArsTeXnica/AT018/UnicodeSC.pdf; zuletzt besucht am 2019-10-02

Literatur/Links (3)

Gosling, P. E.: *PythonTeX Quickstart*; 2016; http://mirrors.ctan.org/macros/latex/contrib/pythontex/pythontex_quickstart.pdf; zuletzt besucht am 2019-10-02

Hilpisch, Yves: *Wissenschaftliches Publizieren mit Python*; 2013;; http://www.hilpisch.com/CAE_Pycon_DE_Scientific_Publishing.pdf; zuletzt besucht am 2019-10-02

Mertz, Andrew; Slough, William: *A Gentle Introduction to PythonTeX*; 2013; hrsg. von Eastern Illinois University; https://tug.org/tug2013/slides/Mertz-A_Gentle_Introduction_to_PythonTeX.pdf; zuletzt besucht am 2019-10-02

Nettles, Bill: *nucleardata – provides nuclide information*; 2018; http://mirrors.ctan.org/macros/latex/contrib/nucleardata/nucleardata.pdf; zuletzt besucht am 2019-10-02

Nettles, Bill; Poore, Geoffrey M.: *Using Python and pdfLaTeX to Generate Customized Physics Problems*; 2016; hrsg. von Union University; https://www.aapt.org/docdirectory/meetingpresentations/WM16/AAPTpaper_CI07_Nettles.pdf; zuletzt besucht am 2019-10-02

Poore, Geoffrey M.: *Reproducible Documents with PythonTeX*; 2013; http://conference.scipy.org/proceedings/scipy2013/pdfs/poore.pdf; zuletzt besucht am 2019-10-02

Literatur/Links (4)

Poore, Geoffrey M.: *PythonTeX – Reproducible documents with LaTeX, Python, and more*; 2015; in *Computational Science & Discovery* 8 (1), S. 1–20; http://iopscience.iop.org/article/10.1088/1749-4699/8/1/014010/pdf; zuletzt besucht am 2019-10-02

Poore, Geoffrey M.: *PythonTeX – Fast Access to Python from within LaTeX*; 2015; http://conference.scipy.org/proceedings/scipy2012/pdfs/geoffrey_poore.pdf; zuletzt besucht am 2019-10-02

Poore, Geoffrey M.: *PythonTeX Gallery*; 2017; http://tug.ctan.org/macros/latex/contrib/pythontex/pythontex_gallery.pdf; zuletzt besucht am 2019-10-02

Poore, Geoffrey M.: *The PythonTeX package – v0.17*; 2019; http://mirrors.ctan.org/macros/latex/contrib/pythontex/pythontex.pdf; zuletzt besucht am 2019-10-02

Benutzte Python-Module

Lehmann, Jörg; Schindler, Michael; Wobst, André: *PyX Manual – Release 0.15.1*; 2019; https://pyx-project.org/; zuletzt besucht am 2019-10-02

Meurer, Aaron; Čertík, Ondřej; Kumar, Amit; Moore, Jason; Singh, Sartaj; Gupta, Harsh: *SymPy Tutorial*; 2016; http://www.sympy.org/scipy-2016-tutorial/intro.pdf; zuletzt besucht am 2019-10-02

PythonTeX: pycode, Schleife, Tabelle, Zweierpotenzen

m	2^m
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024
11	2048
12	4096
13	8192
14	16384
15	32768
16	65536
17	131072
18	262144
19	524288
20	1048576
21	2097152
22	4194304
23	8388608
24	16777216
25	33554432
26	67108864
27	134217728
28	268435456
29	536870912
30	1073741824
31	2147483648
32	4294967296
33	8589934592
34	17179869184
35	34359738368
36	68719476736
37	137438953472
38	274877906944
39	549755813888
40	1099511627776

```latex
% pytex5
% aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% pycode, Schleife, Tabelle, Zweierpotenzen
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}

\begin{document}
\section*{PythonTeX: pycode, Schleife, Tabelle, Zweierpotenzen}

\begin{pycode}
# Aufbau einer tabular-Umgebung in einer Schleife
anfang, ende = 1, 40
print(r"\begin{tabular}{r|r}")
print(r"$m$ & $2^m$ \\ \hline")
for m in range(anfang, ende + 1):    # Schleife 1-40
#    print(r"%d & %d \\" % (m, 2**m))
    print(m, "&", 2**m, r"\\")
print(r"\end{tabular}")
\end{pycode}
\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python5.py
# aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
# Schleife, Tabelle, Zweierpotenzen
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")

print(r"\begin{document}")
print(r"\section*{Python5: Schleife, Tabelle, Zweierpotenzen}")

anfang, ende = 1, 40                # Grenzen
print(r"\begin{tabular}{r|r}")      # tabular-Umgebung in einer Schleife
print(r"$m$ & $2^m$ \\ \hline")     # Tabellenkopf
for m in range(anfang, ende + 1):   # Schleife 1-40
    print(m, "&", 2**m, r"\\")
print(r"\end{tabular}")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\begin{document}
\section*{Python5: Schleife, Tabelle, Zweierpotenzen}
\begin{tabular}{r|r}
$m$ & $2^m$ \\ \hline
1 & 2 \\
2 & 4 \\
3 & 8 \\
4 & 16 \\
5 & 32 \\
6 & 64 \\
7 & 128 \\
8 & 256 \\
9 & 512 \\
10 & 1024 \\
11 & 2048 \\
12 & 4096 \\
13 & 8192 \\
14 & 16384 \\
15 & 32768 \\
16 & 65536 \\
17 & 131072 \\
18 & 262144 \\
19 & 524288 \\
20 & 1048576 \\
21 & 2097152 \\
22 & 4194304 \\
23 & 8388608 \\
24 & 16777216 \\
25 & 33554432 \\
26 & 67108864 \\
27 & 134217728 \\
28 & 268435456 \\
29 & 536870912 \\
30 & 1073741824 \\
31 & 2147483648 \\
32 & 4294967296 \\
33 & 8589934592 \\
34 & 17179869184 \\
35 & 34359738368 \\
36 & 68719476736 \\
37 & 137438953472 \\
38 & 274877906944 \\
```

```
39 & 549755813888 \\
40 & 1099511627776 \\
\end{tabular}
\end{document}
```

PythonTeX: Modul math, pythontexcustomcode, def, Fibonacci, Funktionstabelle, pyc

m	$\sum_{i=1}^{n} i$
1	1
2	3
3	6
4	10
5	15
6	21
7	28
8	36
9	45
10	55
11	66
12	78
13	91
14	105
15	120
16	136
17	153
18	171
19	190
20	210
21	231
22	253
23	276
24	300
25	325
26	351
27	378
28	406
29	435
30	465
31	496
32	528
33	561
34	595
35	630
36	666
37	703
38	741
39	780
40	820

m	F_m
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21
9	34
10	55
11	89
12	144
13	233
14	377
15	610
16	987
17	1597
18	2584
19	4181
20	6765
21	10946
22	17711
23	28657
24	46368
25	75025
26	121393
27	196418
28	317811
29	514229
30	832040
31	1346269
32	2178309
33	3524578
34	5702887
35	9227465
36	14930352
37	24157817
38	39088169
39	63245986
40	102334155

m	$m!$
1	1
2	2
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800
11	39916800
12	479001600
13	6227020800
14	87178291200
15	1307674368000
16	20922789888000
17	355687428096000
18	6402373705728000
19	121645100408832000
20	2432902008176640000
21	51090942171709440000
22	1124000727777607680000
23	25852016738884976640000
24	620448401733239439360000
25	15511210043330985984000000

```
% pytex8
% aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% Modul math, pythontexcustomcode, def, Fibonacci, Funktionstabelle, \pyc
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}

\begin{document}
\section*{PythonTeX: Modul math, pythontexcustomcode, def, Fibonacci, Funktionstabelle, py

\begin{pythontexcustomcode}{py}
import math
def tabelliereFunktion(anfang, ende, funct, label):
    print(r"\begin{tabular}[t]{r|r}")
    print("$m$", "&", label, r"\\ \hline")
    for m in range(anfang, ende + 1):
#        print(r"%d & %d \\" % (m, funct(m)))
        print(m, "&", funct(m), r"\\")
    print(r"\end{tabular}")

def fibo(n):             # rekursive Berechnung des n-ten Fibonacci-Werts
    a, b = 0, 1          # Rekursionsanfang
    for i in range(n):
        a, b = b, a + b  # Rekursionsbedingung
    return a

def summe(n):
    return sum(range(n+1))

\end{pythontexcustomcode}

\pyc{tabelliereFunktion(1, 40, summe, r"$\sum_{i=1}^n i$")}
\pyc{tabelliereFunktion(1, 40, fibo, "$F_{m}$")}
\pyc{tabelliereFunktion(1, 25, math.factorial, "$m!$")}
\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python8.py

import math

def tabelliereFunktion(anfang, ende, funct, label): # tabelliert eine Funktion
    print(r"\begin{tabular}[t]{r|r}")               # tabular-Umgebung
    print("$m$", "&", label, r"\\ \hline")          # Kopfzeile
    for m in range(anfang, ende + 1):               # Schleife
        print(m, "&", funct(m), r"\\")
    print(r"\end{tabular}\hfill")

def fibo(n):                                        # rekursive Berechnung des n-ten Fibon
    a, b = 0, 1                                     # Rekursionsanfang
    for i in range(n):
        a, b = b, a + b                             # Rekursionsbedingung
    return a

def summe(n):                                       # summiert die Zahlen 0, ..., n
    return sum(range(n+1))

print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\begin{document}")
print(r"\section*{python8: Modul math, def, Fibonacci, Funktionstabelle}")

tabelliereFunktion(1, 40, summe, r"$\sum_{i=1}^n i$")
tabelliereFunktion(1, 40, fibo, "$F_{m}$")
tabelliereFunktion(1, 25, math.factorial, "$m!$")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\begin{document}
\section*{python8: Modul math, def, Fibonacci, Funktionstabelle}
\begin{tabular}[t]{r|r}
$m$ & $\sum_{i=1}^n i$ \\ \hline
1 & 1 \\
2 & 3 \\
3 & 6 \\
4 & 10 \\
5 & 15 \\
6 & 21 \\
7 & 28 \\
8 & 36 \\
9 & 45 \\
10 & 55 \\
11 & 66 \\
12 & 78 \\
13 & 91 \\
14 & 105 \\
15 & 120 \\
16 & 136 \\
17 & 153 \\
18 & 171 \\
19 & 190 \\
20 & 210 \\
21 & 231 \\
22 & 253 \\
23 & 276 \\
24 & 300 \\
25 & 325 \\
26 & 351 \\
27 & 378 \\
28 & 406 \\
29 & 435 \\
30 & 465 \\
31 & 496 \\
32 & 528 \\
33 & 561 \\
34 & 595 \\
35 & 630 \\
36 & 666 \\
37 & 703 \\
38 & 741 \\
```

```
39 & 780 \\
40 & 820 \\
\end{tabular}\hfill
\begin{tabular}[t]{r|r}
$m$ & $F_{m}$ \\ \hline
1 & 1 \\
2 & 1 \\
3 & 2 \\
4 & 3 \\
5 & 5 \\
6 & 8 \\
7 & 13 \\
8 & 21 \\
9 & 34 \\
10 & 55 \\
11 & 89 \\
12 & 144 \\
13 & 233 \\
14 & 377 \\
15 & 610 \\
16 & 987 \\
17 & 1597 \\
18 & 2584 \\
19 & 4181 \\
20 & 6765 \\
21 & 10946 \\
22 & 17711 \\
23 & 28657 \\
24 & 46368 \\
25 & 75025 \\
26 & 121393 \\
27 & 196418 \\
28 & 317811 \\
29 & 514229 \\
30 & 832040 \\
31 & 1346269 \\
32 & 2178309 \\
33 & 3524578 \\
34 & 5702887 \\
35 & 9227465 \\
36 & 14930352 \\
37 & 24157817 \\
38 & 39088169 \\
39 & 63245986 \\
```

```
40 & 102334155 \\
\end{tabular}\hfill
\begin{tabular}[t]{r|r}
$m$ & $m!$ \\ \hline
1 & 1 \\
2 & 2 \\
3 & 6 \\
4 & 24 \\
5 & 120 \\
6 & 720 \\
7 & 5040 \\
8 & 40320 \\
9 & 362880 \\
10 & 3628800 \\
11 & 39916800 \\
12 & 479001600 \\
13 & 6227020800 \\
14 & 87178291200 \\
15 & 1307674368000 \\
16 & 20922789888000 \\
17 & 355687428096000 \\
18 & 6402373705728000 \\
19 & 121645100408832000 \\
20 & 2432902008176640000 \\
21 & 51090942171709440000 \\
22 & 1124000727777607680000 \\
23 & 25852016738884976640000 \\
24 & 620448401733239439360000 \\
25 & 15511210043330985984000000 \\
\end{tabular}\hfill
\end{document}
```

PythonTeX: pyblock, printpythontex, pycode, Tabellen, Schleife, $(2,3,4,5)^m$, $m*n$, $m+n$, m^n

Berechne $(2,3,4,5)^m$

```
# Aufbau einer Tabelle mittels einer tabular-Umgebung:
print(r"\begin{tabular}{r|rrrr}")                        # tabular-Umgebung, Anfang
print(r"$m$ & $2^m$ & $3^m$ & $4^m$ & $5^m$ \\ \hline") # Tabellenkopf
for m in range(31):                                      # eine Zeile
    ## print(r"%d & %d \\" % (m, 3**m))
    print(m, "&", 2**m, "&", 3**m, "&", 4**m, "&", 5**m, r"\\")
print(r"\end{tabular}")                                  # tabular-Umgebung, Ende
```

m	2^m	3^m	4^m	5^m
0	1	1	1	1
1	2	3	4	5
2	4	9	16	25
3	8	27	64	125
4	16	81	256	625
5	32	243	1024	3125
6	64	729	4096	15625
7	128	2187	16384	78125
8	256	6561	65536	390625
9	512	19683	262144	1953125
10	1024	59049	1048576	9765625
11	2048	177147	4194304	48828125
12	4096	531441	16777216	244140625
13	8192	1594323	67108864	1220703125
14	16384	4782969	268435456	6103515625
15	32768	14348907	1073741824	30517578125
16	65536	43046721	4294967296	152587890625
17	131072	129140163	17179869184	762939453125
18	262144	387420489	68719476736	3814697265625
19	524288	1162261467	274877906944	19073486328125
20	1048576	3486784401	1099511627776	95367431640625
21	2097152	10460353203	4398046511104	476837158203125
22	4194304	31381059609	17592186044416	2384185791015625
23	8388608	94143178827	70368744177664	11920928955078125
24	16777216	282429536481	281474976710656	59604644775390625
25	33554432	847288609443	1125899906842624	298023223876953125
26	67108864	2541865828329	4503599627370496	1490116119384765625
27	134217728	7625597484987	18014398509481984	7450580596923828125
28	268435456	22876792454961	72057594037927936	37252902984619140625
29	536870912	68630377364883	288230376151711744	186264514923095703125
30	1073741824	205891132094649	1152921504606846976	931322574615478515625

Berechne $m * n$

```
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")    # tabular-Umgebung, Anfang
kopf =  "$m*n$"                            # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + r"$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                        # eine Zeile
```

```
    print(n)
    for m in range(10):
        print(" & ", n*m)
    print(r"\\")
print(r"\end{tabular}")                    # tabular-Umgebung, Ende
```

$m*n$	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

Berechne $m+n$

```
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")    # tabular-Umgebung, Anfang
kopf = "$m+n$"                             # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + r"$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                        # eine Zeile
    print(n)
    for m in range(10):
        print(" & ", n+m)
    print(r"\\")
print(r"\end{tabular}")                    # tabular-Umgebung, Ende
```

$m+n$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18

Berechne m^n

```python
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")    # tabular-Umgebung, Anfang
kopf = "$m^n$"                             # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + r"$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                        # eine Zeile
    print(n)
    for m in range(10):
        print(" & ", n**m)
    print(r"\\")
print(r"\end{tabular}")                    # tabular-Umgebung, Ende
```

m^n	0	1	2	3	4	5	6	7	8	9
0	1	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
2	1	2	4	8	16	32	64	128	256	512
3	1	3	9	27	81	243	729	2187	6561	19683
4	1	4	16	64	256	1024	4096	16384	65536	262144
5	1	5	25	125	625	3125	15625	78125	390625	1953125
6	1	6	36	216	1296	7776	46656	279936	1679616	10077696
7	1	7	49	343	2401	16807	117649	823543	5764801	40353607
8	1	8	64	512	4096	32768	262144	2097152	16777216	134217728
9	1	9	81	729	6561	59049	531441	4782969	43046721	387420489

```
% pytex20
%%% pyblock, printpythontex, pycode, Tabellen, Schleife, $(2,3,4,5)^m$, $m*n$, $m+n$, $m^n
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}

\begin{document}
\section*{PythonTeX: pyblock, printpythontex, pycode, Tabellen, Schleife, $(2,3,4,5)^m$, $

\subsection*{Berechne $(2,3,4,5)^m$}
\begin{pyblock}[xhochm]
# Aufbau einer Tabelle mittels einer tabular-Umgebung:
print(r"\begin{tabular}{r|rrrr}")                        # tabular-Umgebung, Anfang
print(r"$m$ & $2^m$ & $3^m$ & $4^m$ & $5^m$ \\ \hline") # Tabellenkopf
for m in range(31):                                      # eine Zeile
    ## print(r"%d & %d \\" % (m, 3**m))
    print(m, "&", 2**m, "&", 3**m, "&", 4**m, "&", 5**m, r"\\")
print(r"\end{tabular}")                                  # tabular-Umgebung, Ende
\end{pyblock}
\printpythontex

\subsection*{Berechne $m*n$}
\begin{pyblock}[mult]
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")   # tabular-Umgebung, Anfang
kopf =  "$m*n$"                           # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + r"$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                       # eine Zeile
    print(n)
    for m in range(10):
        print(" & ", n*m)
    print(r"\\")
print(r"\end{tabular}")                   # tabular-Umgebung, Ende
\end{pyblock}
\printpythontex

\subsection*{Berechne $m+n$}
\begin{pyblock}[add]
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")   # tabular-Umgebung, Anfang
```

```
kopf = "$m+n$"                              # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + r"$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                         # eine Zeile
    print(n)
    for m in range(10):
        print(" & ", n+m)
    print(r"\\")
print(r"\end{tabular}")                     # tabular-Umgebung, Ende
\end{pyblock}
\printpythontex

\subsection*{Berechne $m^n$}
\begin{pyblock}[pot]
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")    # tabular-Umgebung, Anfang
kopf = "$m^n$"                              # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + r"$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                         # eine Zeile
    print(n)
    for m in range(10):
        print(" & ", n**m)
    print(r"\\")
print(r"\end{tabular}")                     # tabular-Umgebung, Ende
\end{pyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python20.py
# Tabellen, Schleife, $(2,3,4,5)^m$, $m*n$, $m+n$, $m^n$, session
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")

print(r"\begin{document}")
print(r"\section*{Python20: Tabellen, Schleife, $(2,3,4,5)^m$, $m*n$, $m+n$, $m^n$}")

print(r"\subsection*{Berechne $(2,3,4,5)^m$}")
# Aufbau einer Tabelle mittels einer tabular-Umgebung:
print(r"\begin{tabular}{r|rrrr}")                        # tabular-Umgebung, Anfang
print(r"$m$ & $2^m$ & $3^m$ & $4^m$ & $5^m$ \\ \hline") # Tabellenkopf
for m in range(31):                                      # eine Zeile
    print(m, "&", 2**m, "&", 3**m, "&", 4**m, "&", 5**m, r"\\")
print(r"\end{tabular}")                                  # tabular-Umgebung, Ende

print(r"\subsection*{Berechne $m*n$}")
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")                  # tabular-Umgebung, Anfang
kopf = "$m*n$"                                           # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + "$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                                      # eine Zeile
    print(n)
    for m in range(10):
        print(" & ", n * m)
    print(r"\\")
print(r"\end{tabular}")                                  # tabular-Umgebung, Ende

print(r"\subsection*{Berechne $m+n$}")
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")                  # tabular-Umgebung, Anfang
kopf = "$m+n$"                                           # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + "$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):  # eine Zeile
    print(n)
    for m in range(10):
```

```
        print(" & ", n + m)
    print(r"\\")
print(r"\end{tabular}")                                  # tabular-Umgebung, Ende

print(r"\subsection*{Berechne $m^n$}")
# Aufbau einer Tabelle mittels einer tabular-Umgebung
print(r"\begin{tabular}{r|rrrrrrrrrr}")                  # tabular-Umgebung, Anfang
kopf = "$m^n$"                                           # Tabellenkopf
for m in range(10):
    kopf += " & $" + str(m) + "$"
kopf += r"\\ \hline"
print(kopf)
for n in range(10):                                      # eine Zeile
    print(n)
    for m in range(10):
        print(" & ", n ** m)
    print(r"\\")
print(r"\end{tabular}")                                  # tabular-Umgebung, Ende

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\begin{document}
\section*{Python20: Tabellen, Schleife, $(2,3,4,5)^m$, $m*n$, $m+n$, $m^n$}
\subsection*{Berechne $(2,3,4,5)^m$}
\begin{tabular}{r|rrrr}
$m$ & $2^m$ & $3^m$ & $4^m$ & $5^m$ \\ \hline
0 & 1 & 1 & 1 & 1 \\
1 & 2 & 3 & 4 & 5 \\
2 & 4 & 9 & 16 & 25 \\
3 & 8 & 27 & 64 & 125 \\
4 & 16 & 81 & 256 & 625 \\
5 & 32 & 243 & 1024 & 3125 \\
6 & 64 & 729 & 4096 & 15625 \\
7 & 128 & 2187 & 16384 & 78125 \\
8 & 256 & 6561 & 65536 & 390625 \\
9 & 512 & 19683 & 262144 & 1953125 \\
10 & 1024 & 59049 & 1048576 & 9765625 \\
11 & 2048 & 177147 & 4194304 & 48828125 \\
12 & 4096 & 531441 & 16777216 & 244140625 \\
13 & 8192 & 1594323 & 67108864 & 1220703125 \\
14 & 16384 & 4782969 & 268435456 & 6103515625 \\
15 & 32768 & 14348907 & 1073741824 & 30517578125 \\
16 & 65536 & 43046721 & 4294967296 & 152587890625 \\
17 & 131072 & 129140163 & 17179869184 & 762939453125 \\
18 & 262144 & 387420489 & 68719476736 & 3814697265625 \\
19 & 524288 & 1162261467 & 274877906944 & 19073486328125 \\
20 & 1048576 & 3486784401 & 1099511627776 & 95367431640625 \\
21 & 2097152 & 10460353203 & 4398046511104 & 476837158203125 \\
22 & 4194304 & 31381059609 & 17592186044416 & 2384185791015625 \\
23 & 8388608 & 94143178827 & 70368744177664 & 11920928955078125 \\
24 & 16777216 & 282429536481 & 281474976710656 & 59604644775390625 \\
25 & 33554432 & 847288609443 & 1125899906842624 & 298023223876953125 \\
26 & 67108864 & 2541865828329 & 4503599627370496 & 1490116119384765625 \\
27 & 134217728 & 7625597484987 & 18014398509481984 & 7450580596923828125 \\
28 & 268435456 & 22876792454961 & 72057594037927936 & 37252902984619140625 \\
29 & 536870912 & 68630377364883 & 288230376151711744 & 186264514923095703125 \\
30 & 1073741824 & 205891132094649 & 1152921504606846976 & 931322574615478515625 \\
\end{tabular}
\subsection*{Berechne $m*n$}
\begin{tabular}{r|rrrrrrrrrr}
$m*n$ & $0$ & $1$ & $2$ & $3$ & $4$ & $5$ & $6$ & $7$ & $8$ & $9$\\ \hline
0
 &  0
```

```
 &   0
 &   0
 &   0
 &   0
 &   0
 &   0
 &   0
 &   0
 &   0
\\
1
 &   0
 &   1
 &   2
 &   3
 &   4
 &   5
 &   6
 &   7
 &   8
 &   9
\\
2
 &   0
 &   2
 &   4
 &   6
 &   8
 &   10
 &   12
 &   14
 &   16
 &   18
\\
3
 &   0
 &   3
 &   6
 &   9
 &   12
 &   15
 &   18
 &   21
 &   24
```

```
 &   27
\\
4
 &   0
 &   4
 &   8
 &   12
 &   16
 &   20
 &   24
 &   28
 &   32
 &   36
\\
5
 &   0
 &   5
 &   10
 &   15
 &   20
 &   25
 &   30
 &   35
 &   40
 &   45
\\
6
 &   0
 &   6
 &   12
 &   18
 &   24
 &   30
 &   36
 &   42
 &   48
 &   54
\\
7
 &   0
 &   7
 &   14
 &   21
 &   28
```

```
 &  35
 &  42
 &  49
 &  56
 &  63
\\
8
 &  0
 &  8
 &  16
 &  24
 &  32
 &  40
 &  48
 &  56
 &  64
 &  72
\\
9
 &  0
 &  9
 &  18
 &  27
 &  36
 &  45
 &  54
 &  63
 &  72
 &  81
\\
\end{tabular}
\subsection*{Berechne $m+n$}
\begin{tabular}{r|rrrrrrrrrr}
$m+n$ & $0$ & $1$ & $2$ & $3$ & $4$ & $5$ & $6$ & $7$ & $8$ & $9$\\ \hline
0
 &  0
 &  1
 &  2
 &  3
 &  4
 &  5
 &  6
 &  7
 &  8
```

```
 &   9
\\
1
 &   1
 &   2
 &   3
 &   4
 &   5
 &   6
 &   7
 &   8
 &   9
 &   10
\\
2
 &   2
 &   3
 &   4
 &   5
 &   6
 &   7
 &   8
 &   9
 &   10
 &   11
\\
3
 &   3
 &   4
 &   5
 &   6
 &   7
 &   8
 &   9
 &   10
 &   11
 &   12
\\
4
 &   4
 &   5
 &   6
 &   7
 &   8
```

```
 &   9
 &   10
 &   11
 &   12
 &   13
\\
5
 &   5
 &   6
 &   7
 &   8
 &   9
 &   10
 &   11
 &   12
 &   13
 &   14
\\
6
 &   6
 &   7
 &   8
 &   9
 &   10
 &   11
 &   12
 &   13
 &   14
 &   15
\\
7
 &   7
 &   8
 &   9
 &   10
 &   11
 &   12
 &   13
 &   14
 &   15
 &   16
\\
8
 &   8
```

```
 &  9
 &  10
 &  11
 &  12
 &  13
 &  14
 &  15
 &  16
 &  17
\\
9
 &  9
 &  10
 &  11
 &  12
 &  13
 &  14
 &  15
 &  16
 &  17
 &  18
\\
\end{tabular}
\subsection*{Berechne $m^n$}
\begin{tabular}{r|rrrrrrrrrr}
$m^n$ & $0$ & $1$ & $2$ & $3$ & $4$ & $5$ & $6$ & $7$ & $8$ & $9$\\ \hline
0
 &  1
 &  0
 &  0
 &  0
 &  0
 &  0
 &  0
 &  0
 &  0
 &  0
\\
1
 &  1
 &  1
 &  1
 &  1
 &  1
```

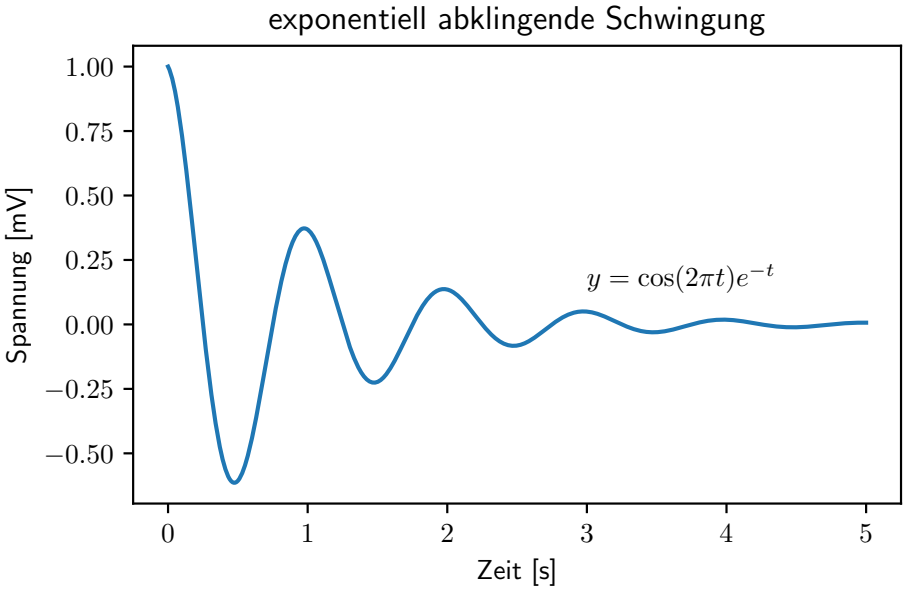
```
 &   1
 &   1
 &   1
 &   1
 &   1
\\
2
 &   1
 &   2
 &   4
 &   8
 &   16
 &   32
 &   64
 &   128
 &   256
 &   512
\\
3
 &   1
 &   3
 &   9
 &   27
 &   81
 &   243
 &   729
 &   2187
 &   6561
 &   19683
\\
4
 &   1
 &   4
 &   16
 &   64
 &   256
 &   1024
 &   4096
 &   16384
 &   65536
 &   262144
\\
5
 &   1
```

```
 &   5
 &   25
 &   125
 &   625
 &   3125
 &   15625
 &   78125
 &   390625
 &   1953125
\\
6
 &   1
 &   6
 &   36
 &   216
 &   1296
 &   7776
 &   46656
 &   279936
 &   1679616
 &   10077696
\\
7
 &   1
 &   7
 &   49
 &   343
 &   2401
 &   16807
 &   117649
 &   823543
 &   5764801
 &   40353607
\\
8
 &   1
 &   8
 &   64
 &   512
 &   4096
 &   32768
 &   262144
 &   2097152
 &   16777216
```

```latex
 &  134217728
\\
9
 &  1
 &  9
 &  81
 &  729
 &  6561
 &  59049
 &  531441
 &  4782969
 &  43046721
 &  387420489
\\
\end{tabular}
\end{document}
```

PythonTeX: pyblock, printpythontex, Grafik mit pylab, Graph einer Funktion

```python
from pylab import *

def f(t):                                          # (1) Funktion f(t) definieren
    return cos(2 * pi * t) * exp(-t)

t = linspace(0, 5, 500)                            # (2) auszugebende Punkte berechnen
y = f(t)

clf()                                              # (3) Anfang: leere Grafik (5in x 3in)
figure(figsize=(5, 3))

rc("text", usetex=True)                            # (4) TeX-Schriften benutzen

#                                                  # (5) eigentliche Grafik (+ Anmerkungen)
plot(t, y)                                         # Funktion ausgeben
title("exponentiell abklingende Schwingung")       # Kopftitel
text(3, 0.15, r"$y = \cos(2 \pi t) e^{-t}$")       # Text bei (3, 0.15)
xlabel("Zeit [s]")                                 # Beschriftung der x-Achse
ylabel("Spannung [mV]")                            # Beschriftung der y-Achse

savefig("meinegrafik.pdf", bbox_inches="tight") # (6) Grafik als PDF-Datei speichern

print(r"\begin{center}")                           # (7) Grafik im LaTeX-Dokument ausgeben
print(r"\includegraphics[width=0.85\textwidth]{meinegrafik.pdf}")
print(r"\end{center}")
```


exponentiell abklingende Schwingung
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
Spannung [mV]
$y = \cos(2\pi t)e^{-t}$
0
1
2
3
4
5
Zeit [s]

```latex
% pytex38
% aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% pyblock, \printpythontex, Grafik mit pylab, Graph einer Funktion
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{graphicx}

\begin{document}
\section*{PythonTeX: pyblock, printpythontex, Grafik mit pylab, Graph einer Funktion}

\begin{pyblock}
from pylab import *

def f(t):                                          # (1) Funktion f(t) definieren
    return cos(2 * pi * t) * exp(-t)

t = linspace(0, 5, 500)                            # (2) auszugebende Punkte berechnen
y = f(t)

clf()                                              # (3) Anfang: leere Grafik (5in x 3in)
figure(figsize=(5, 3))

rc("text", usetex=True)                            # (4) TeX-Schriften benutzen

#                                                  # (5) eigentliche Grafik (+ Anmerkungen)
plot(t, y)                                         # Funktion ausgeben
title("exponentiell abklingende Schwingung")       # Kopftitel
text(3, 0.15, r"$y = \cos(2 \pi t) e^{-t}$")       # Text bei (3, 0.15)
xlabel("Zeit [s]")                                 # Beschriftung der x-Achse
ylabel("Spannung [mV]")                            # Beschriftung der y-Achse

savefig("meinegrafik.pdf", bbox_inches="tight") # (6) Grafik als PDF-Datei speichern

print(r"\begin{center}")                           # (7) Grafik im LaTeX-Dokument ausgeben
print(r"\includegraphics[width=0.85\textwidth]{meinegrafik.pdf}")
print(r"\end{center}")
\end{pyblock}
\printpythontex
\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python38.py
# aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
# pyblock, Grafik mit pylab, Graph einer Funktion
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{graphicx}")

print(r"\begin{document}")
print(r"\section*{Python38: Grafik mit pylab, Graph einer Funktion}")

from pylab import *

def f(t):                                       # (1) Funktion f(t) definieren
    return cos(2 * pi * t) * exp(-t)

t = linspace(0, 5, 500)                         # (2) auszugebende Punkte berechnen
y = f(t)

clf()                                           # (3) Anfang: leere Grafik (5 x 3 Zoll)
figure(figsize=(5, 3))

rc("text", usetex=True)                         # (4) TeX-Schriften benutzen

#                                               # (5) eigentliche Grafik (+ Anmerkungen)
plot(t, y)                                      # Funktion ausgeben
title("exponentiell abklingende Schwingung")    # Kopftitel
text(3, 0.15, r"$y = \cos(2 \pi t) e^{-t}$")    # Text bei (3, 0.15)
xlabel("Zeit [s]")                              # Beschriftung der x-Achse
ylabel("Spannung [mV]")                         # Beschriftung der y-Achse

savefig("meinegrafik.pdf", bbox_inches="tight") # (6) Grafik als PDF-Datei speichern

print(r"\begin{center}")                        # (7) Grafik im LaTeX-Dokument ausgeben
print(r"\includegraphics[width=0.85\textwidth]{meinegrafik.pdf}")
print(r"\end{center}")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{graphicx}
\begin{document}
\section*{Python38: Grafik mit pylab, Graph einer Funktion}
\begin{center}
\includegraphics[width=0.85\textwidth]{meinegrafik.pdf}
\end{center}
\end{document}
```

PythonTeX: pyblock, Grafik mit PyX, Graph einer Funktion

```
from pyx import *                                # PyX importieren
#text.set(fontmap="psfonts.map psfonts.cmz psfonts.amz")

g = graph.graphxy(width=10)                      # neue Grafik, Größe der Grafik
g.plot(graph.data.function("y(x) = sin(x) / x", min=-20, max=20))
g.writePDFfile("func1")                          # Grafik als PDF-Datei

g = graph.graphxy(width=10)                      # Größe der Grafik
g.plot(graph.data.function("y(x) = cos(x) / x", min=-20, max=20))
g.writePDFfile("func2")                          # Grafik als PDF-Datei
```

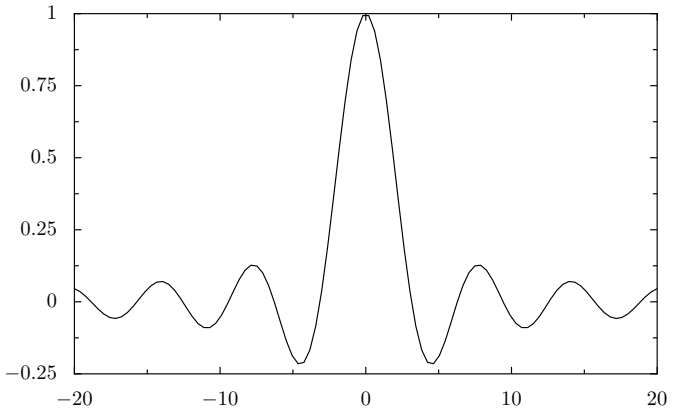


Abbildung 1: $y(x) = \frac{\sin(x)}{x}$

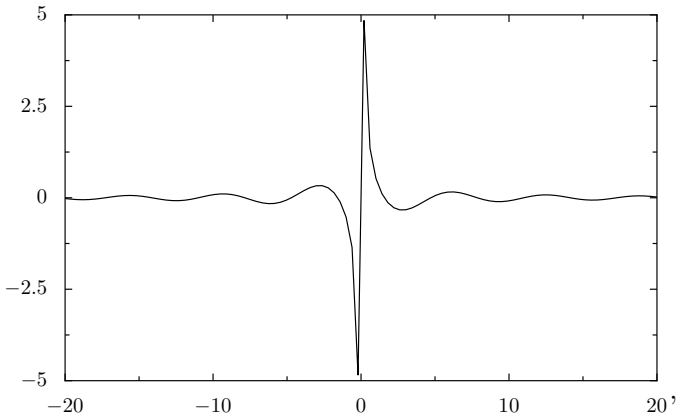


Abbildung 2: $y(x) = \frac{\cos(x)}{x}$

```latex
% pytex23
% nach Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% pyblock, Modul sympy, Funktionen, Ableitungen, Integrale, derivative, latex, eval
\documentclass[parskip=half,fontsize=10,DIV=9,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}      % Mathematik à la AMS

\begin{document}
\section*{PythonTeX: pyblock, sympy, Funktionen, Ableitungen, Integrale}

\begin{pyblock}
from sympy import *                             # symbolische Mathematik
var('x')                                        # symbolische Variable x

# Liste von Funktionen, die in der Tabelle aufgeführt werden sollen
funktionen = ['acos(x)', 'acosh(x)', 'acot(x)', 'acoth(x)', 'asin(x)', 'asinh(x)',
             'atan(x)', 'atanh(x)', 'cos(x)**2', 'cos(x)', 'cosh(x)', 'cot(x)**2',
             'cot(x)', 'coth(x)', 'erf(x)', 'erfc(x)', 'exp(x)', 'gamma(x)','ln(x)',
             'sin(x)**2', 'sin(x)', 'sinh(x)', 'sqrt(x)', 'tan(x)**2', 'tan(x)',
             'tanh(x)', 'csc(x)']

f          = "f(x) "
fableit    = "f'(x)"
fint       = r"\int\!\!f(x)\,dx"

print(r'\begin{align*}')                          # Tabelle
for funk in funktionen:                           # Schleife
    if funk in ['sin(x)**2']:                     # ggf. neue Seite
        print(r"\displaybreak")
    meineableit = 'Derivative(' + funk + ', x)'   # Ableitung
    meinint     = 'Integral(' + funk + ', x)'     # Integral
    print(latex(f),       r"&=", latex(eval(funk)),                    r'&')
    print(latex(fableit), r"&=", latex(eval(meineableit + '.doit()')), r"&")
    print(latex(fint),    r"&=", latex(eval(meinint + '.doit()')),     r'\\')
print(r'\end{align*}')
\end{pyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python22.tex
# pyblock, Modul pyx, Graph einer Funktion
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{graphicx}    % Einbinden von Graphiken")
print(r"\usepackage[english, ngerman]{babel}")

print(r"\begin{document}")

print(r"\section*{Python22: Modul pyx, Graph einer Funktion}")

from pyx import *                           # PyX importieren

g = graph.graphxy(width=10)                 # neue Grafik, Größe der Grafik
g.plot(graph.data.function("y(x) = sin(x) / x", min=-20, max=20))
g.writePDFfile("func1")                     # Grafik als PDF-Datei

g = graph.graphxy(width=10)                 # Größe der Grafik
g.plot(graph.data.function("y(x) = cos(x) / x", min=-20, max=20))
g.writePDFfile("func2")                     # Grafik als PDF-Datei

print(r"\begin{figure}[!ht]")
print(r"\centering")
print(r"\includegraphics[width=0.6\textwidth]{func1}")
print(r"\caption{$y(x)=\frac{\sin(x)}{x}$}")
print(r"\end{figure}")

print(r"\begin{figure}[!ht]")
print(r"\centering")
print(r"\includegraphics[width=0.6\textwidth]{func2}")
print(r"\caption{$y(x)=\frac{\cos(x)}{x}$}")
print(r"\end{figure}")
print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{graphicx}     % Einbinden von Graphiken
\usepackage[english, ngerman]{babel}
\begin{document}
\section*{Python22: Modul pyx, Graph einer Funktion}
\begin{figure}[!ht]
\centering
\includegraphics[width=0.6\textwidth]{func1}
\caption{$y(x)=\frac{\sin(x)}{x}$}
\end{figure}
\begin{figure}[!ht]
\centering
\includegraphics[width=0.6\textwidth]{func2}
\caption{$y(x)=\frac{\cos(x)}{x}$}
\end{figure}
\end{document}
```

PythonTeX: pythontexcustomcode, sympy, def, Schleife, Primzahl

Die ersten 500 Primzahlen sind 2 , 3 , 5 , 7 , 11 , 13 , 17 , 19 , 23 , 29 , 31 , 37 , 41 , 43 , 47 , 53 , 59 , 61 , 67 , 71 , 73 , 79 , 83 , 89 , 97 , 101 , 103 , 107 , 109 , 113 , 127 , 131 , 137 , 139 , 149 , 151 , 157 , 163 , 167 , 173 , 179 , 181 , 191 , 193 , 197 , 199 , 211 , 223 , 227 , 229 , 233 , 239 , 241 , 251 , 257 , 263 , 269 , 271 , 277 , 281 , 283 , 293 , 307 , 311 , 313 , 317 , 331 , 337 , 347 , 349 , 353 , 359 , 367 , 373 , 379 , 383 , 389 , 397 , 401 , 409 , 419 , 421 , 431 , 433 , 439 , 443 , 449 , 457 , 461 , 463 , 467 , 479 , 487 , 491 , 499 , 503 , 509 , 521 , 523 , 541 , 547 , 557 , 563 , 569 , 571 , 577 , 587 , 593 , 599 , 601 , 607 , 613 , 617 , 619 , 631 , 641 , 643 , 647 , 653 , 659 , 661 , 673 , 677 , 683 , 691 , 701 , 709 , 719 , 727 , 733 , 739 , 743 , 751 , 757 , 761 , 769 , 773 , 787 , 797 , 809 , 811 , 821 , 823 , 827 , 829 , 839 , 853 , 857 , 859 , 863 , 877 , 881 , 883 , 887 , 907 , 911 , 919 , 929 , 937 , 941 , 947 , 953 , 967 , 971 , 977 , 983 , 991 , 997 , 1009 , 1013 , 1019 , 1021 , 1031 , 1033 , 1039 , 1049 , 1051 , 1061 , 1063 , 1069 , 1087 , 1091 , 1093 , 1097 , 1103 , 1109 , 1117 , 1123 , 1129 , 1151 , 1153 , 1163 , 1171 , 1181 , 1187 , 1193 , 1201 , 1213 , 1217 , 1223 , 1229 , 1231 , 1237 , 1249 , 1259 , 1277 , 1279 , 1283 , 1289 , 1291 , 1297 , 1301 , 1303 , 1307 , 1319 , 1321 , 1327 , 1361 , 1367 , 1373 , 1381 , 1399 , 1409 , 1423 , 1427 , 1429 , 1433 , 1439 , 1447 , 1451 , 1453 , 1459 , 1471 , 1481 , 1483 , 1487 , 1489 , 1493 , 1499 , 1511 , 1523 , 1531 , 1543 , 1549 , 1553 , 1559 , 1567 , 1571 , 1579 , 1583 , 1597 , 1601 , 1607 , 1609 , 1613 , 1619 , 1621 , 1627 , 1637 , 1657 , 1663 , 1667 , 1669 , 1693 , 1697 , 1699 , 1709 , 1721 , 1723 , 1733 , 1741 , 1747 , 1753 , 1759 , 1777 , 1783 , 1787 , 1789 , 1801 , 1811 , 1823 , 1831 , 1847 , 1861 , 1867 , 1871 , 1873 , 1877 , 1879 , 1889 , 1901 , 1907 , 1913 , 1931 , 1933 , 1949 , 1951 , 1973 , 1979 , 1987 , 1993 , 1997 , 1999 , 2003 , 2011 , 2017 , 2027 , 2029 , 2039 , 2053 , 2063 , 2069 , 2081 , 2083 , 2087 , 2089 , 2099 , 2111 , 2113 , 2129 , 2131 , 2137 , 2141 , 2143 , 2153 , 2161 , 2179 , 2203 , 2207 , 2213 , 2221 , 2237 , 2239 , 2243 , 2251 , 2267 , 2269 , 2273 , 2281 , 2287 , 2293 , 2297 , 2309 , 2311 , 2333 , 2339 , 2341 , 2347 , 2351 , 2357 , 2371 , 2377 , 2381 , 2383 , 2389 , 2393 , 2399 , 2411 , 2417 , 2423 , 2437 , 2441 , 2447 , 2459 , 2467 , 2473 , 2477 , 2503 , 2521 , 2531 , 2539 , 2543 , 2549 , 2551 , 2557 , 2579 , 2591 , 2593 , 2609 , 2617 , 2621 , 2633 , 2647 , 2657 , 2659 , 2663 , 2671 , 2677 , 2683 , 2687 , 2689 , 2693 , 2699 , 2707 , 2711 , 2713 , 2719 , 2729 , 2731 , 2741 , 2749 , 2753 , 2767 , 2777 , 2789 , 2791 , 2797 , 2801 , 2803 , 2819 , 2833 , 2837 , 2843 , 2851 , 2857 , 2861 , 2879 , 2887 , 2897 , 2903 , 2909 , 2917 , 2927 , 2939 , 2953 , 2957 , 2963 , 2969 , 2971 , 2999 , 3001 , 3011 , 3019 , 3023 , 3037 , 3041 , 3049 , 3061 , 3067 , 3079 , 3083 , 3089 , 3109 , 3119 , 3121 , 3137 , 3163 , 3167 , 3169 , 3181 , 3187 , 3191 , 3203 , 3209 , 3217 , 3221 , 3229 , 3251 , 3253 , 3257 , 3259 , 3271 , 3299 , 3301 , 3307 , 3313 , 3319 , 3323 , 3329 , 3331 , 3343 , 3347 , 3359 , 3361 , 3371 , 3373 , 3389 , 3391 , 3407 , 3413 , 3433 , 3449 , 3457 , 3461 , 3463 , 3467 , 3469 , 3491 , 3499 , 3511 , 3517 , 3527 , 3529 , 3533 , 3539 , 3541 , 3547 , 3557 , 3559 , und 3571 .

```latex
% pytex11
% aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% sympy, Primzahlen, pythontexcustomcode, def, Schleife
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}

\begin{document}
\section*{PythonTeX: pythontexcustomcode, sympy, def, Schleife, Primzahl}

\begin{pythontexcustomcode}{py}
from sympy import prime              # symbolische Mathematik, hier Primzahlen
n = 500

def Primzahlen(n):                   # Annahme n >= 3
    for i in range(1, n):
#        print("%d, " % prime(i))
#    print("und %d%%" % prime(n))
        print(prime(i), ",")
    print("und", prime(n))
\end{pythontexcustomcode}

Die ersten \py{n} Primzahlen sind \pyc{Primzahlen(n)}.
\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python11
# aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")

print(r"\begin{document}")
print(r"\section*{python11: sympy, def, Schleife, Primzahl}")

from sympy import prime              # symbolische Mathematik, hier Primzahlen
n = 500

def Primzahlen(n):                   # Annahme n >= 3
    for i in range(1, n):
        print(prime(i), ",")
    print("und", prime(n))

print("Die ersten", n, "Primzahlen sind:")
Primzahlen(n)
print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\begin{document}
\section*{python11: sympy, def, Schleife, Primzahl}
Die ersten 500 Primzahlen sind:
2 ,
3 ,
5 ,
7 ,
11 ,
13 ,
17 ,
19 ,
23 ,
29 ,
31 ,
37 ,
41 ,
43 ,
47 ,
53 ,
59 ,
61 ,
67 ,
71 ,
73 ,
79 ,
83 ,
89 ,
97 ,
101 ,
103 ,
107 ,
109 ,
113 ,
127 ,
131 ,
137 ,
139 ,
149 ,
151 ,
157 ,
163 ,
167 ,
```

```
173 ,
179 ,
181 ,
191 ,
193 ,
197 ,
199 ,
211 ,
223 ,
227 ,
229 ,
233 ,
239 ,
241 ,
251 ,
257 ,
263 ,
269 ,
271 ,
277 ,
281 ,
283 ,
293 ,
307 ,
311 ,
313 ,
317 ,
331 ,
337 ,
347 ,
349 ,
353 ,
359 ,
367 ,
373 ,
379 ,
383 ,
389 ,
397 ,
401 ,
409 ,
419 ,
421 ,
431 ,
```

433 ,
439 ,
443 ,
449 ,
457 ,
461 ,
463 ,
467 ,
479 ,
487 ,
491 ,
499 ,
503 ,
509 ,
521 ,
523 ,
541 ,
547 ,
557 ,
563 ,
569 ,
571 ,
577 ,
587 ,
593 ,
599 ,
601 ,
607 ,
613 ,
617 ,
619 ,
631 ,
641 ,
643 ,
647 ,
653 ,
659 ,
661 ,
673 ,
677 ,
683 ,
691 ,
701 ,
709 ,

```
719 ,
727 ,
733 ,
739 ,
743 ,
751 ,
757 ,
761 ,
769 ,
773 ,
787 ,
797 ,
809 ,
811 ,
821 ,
823 ,
827 ,
829 ,
839 ,
853 ,
857 ,
859 ,
863 ,
877 ,
881 ,
883 ,
887 ,
907 ,
911 ,
919 ,
929 ,
937 ,
941 ,
947 ,
953 ,
967 ,
971 ,
977 ,
983 ,
991 ,
997 ,
1009 ,
1013 ,
1019 ,
```

1021 ,
1031 ,
1033 ,
1039 ,
1049 ,
1051 ,
1061 ,
1063 ,
1069 ,
1087 ,
1091 ,
1093 ,
1097 ,
1103 ,
1109 ,
1117 ,
1123 ,
1129 ,
1151 ,
1153 ,
1163 ,
1171 ,
1181 ,
1187 ,
1193 ,
1201 ,
1213 ,
1217 ,
1223 ,
1229 ,
1231 ,
1237 ,
1249 ,
1259 ,
1277 ,
1279 ,
1283 ,
1289 ,
1291 ,
1297 ,
1301 ,
1303 ,
1307 ,
1319 ,

1321 ,
1327 ,
1361 ,
1367 ,
1373 ,
1381 ,
1399 ,
1409 ,
1423 ,
1427 ,
1429 ,
1433 ,
1439 ,
1447 ,
1451 ,
1453 ,
1459 ,
1471 ,
1481 ,
1483 ,
1487 ,
1489 ,
1493 ,
1499 ,
1511 ,
1523 ,
1531 ,
1543 ,
1549 ,
1553 ,
1559 ,
1567 ,
1571 ,
1579 ,
1583 ,
1597 ,
1601 ,
1607 ,
1609 ,
1613 ,
1619 ,
1621 ,
1627 ,
1637 ,

1657 ,
1663 ,
1667 ,
1669 ,
1693 ,
1697 ,
1699 ,
1709 ,
1721 ,
1723 ,
1733 ,
1741 ,
1747 ,
1753 ,
1759 ,
1777 ,
1783 ,
1787 ,
1789 ,
1801 ,
1811 ,
1823 ,
1831 ,
1847 ,
1861 ,
1867 ,
1871 ,
1873 ,
1877 ,
1879 ,
1889 ,
1901 ,
1907 ,
1913 ,
1931 ,
1933 ,
1949 ,
1951 ,
1973 ,
1979 ,
1987 ,
1993 ,
1997 ,
1999 ,

2003 ,
2011 ,
2017 ,
2027 ,
2029 ,
2039 ,
2053 ,
2063 ,
2069 ,
2081 ,
2083 ,
2087 ,
2089 ,
2099 ,
2111 ,
2113 ,
2129 ,
2131 ,
2137 ,
2141 ,
2143 ,
2153 ,
2161 ,
2179 ,
2203 ,
2207 ,
2213 ,
2221 ,
2237 ,
2239 ,
2243 ,
2251 ,
2267 ,
2269 ,
2273 ,
2281 ,
2287 ,
2293 ,
2297 ,
2309 ,
2311 ,
2333 ,
2339 ,
2341 ,

2347 ,
2351 ,
2357 ,
2371 ,
2377 ,
2381 ,
2383 ,
2389 ,
2393 ,
2399 ,
2411 ,
2417 ,
2423 ,
2437 ,
2441 ,
2447 ,
2459 ,
2467 ,
2473 ,
2477 ,
2503 ,
2521 ,
2531 ,
2539 ,
2543 ,
2549 ,
2551 ,
2557 ,
2579 ,
2591 ,
2593 ,
2609 ,
2617 ,
2621 ,
2633 ,
2647 ,
2657 ,
2659 ,
2663 ,
2671 ,
2677 ,
2683 ,
2687 ,
2689 ,

2693 ,
2699 ,
2707 ,
2711 ,
2713 ,
2719 ,
2729 ,
2731 ,
2741 ,
2749 ,
2753 ,
2767 ,
2777 ,
2789 ,
2791 ,
2797 ,
2801 ,
2803 ,
2819 ,
2833 ,
2837 ,
2843 ,
2851 ,
2857 ,
2861 ,
2879 ,
2887 ,
2897 ,
2903 ,
2909 ,
2917 ,
2927 ,
2939 ,
2953 ,
2957 ,
2963 ,
2969 ,
2971 ,
2999 ,
3001 ,
3011 ,
3019 ,
3023 ,
3037 ,

3041 ,
3049 ,
3061 ,
3067 ,
3079 ,
3083 ,
3089 ,
3109 ,
3119 ,
3121 ,
3137 ,
3163 ,
3167 ,
3169 ,
3181 ,
3187 ,
3191 ,
3203 ,
3209 ,
3217 ,
3221 ,
3229 ,
3251 ,
3253 ,
3257 ,
3259 ,
3271 ,
3299 ,
3301 ,
3307 ,
3313 ,
3319 ,
3323 ,
3329 ,
3331 ,
3343 ,
3347 ,
3359 ,
3361 ,
3371 ,
3373 ,
3389 ,
3391 ,
3407 ,

```
3413 ,
3433 ,
3449 ,
3457 ,
3461 ,
3463 ,
3467 ,
3469 ,
3491 ,
3499 ,
3511 ,
3517 ,
3527 ,
3529 ,
3533 ,
3539 ,
3541 ,
3547 ,
3557 ,
3559 ,
und 3571
\end{document}
```

PythonTeX: pyblock, printpythontex, sympy, Binome, Schleife, Tabelle

```python
from sympy import *            # symbolische Mathematik
var("a, b")                    # sympy-Variable
Binome = []                    # Liste für Binomi-Ausdrücke vorbesetzt

for m in range(1, 10):
    Binome.append((a + b)**m) # Binomi-Ausdrücke erzeugen

# Aufbau einer Tabelle mit einer align*-Umgebung
print(r"\begin{align*}")
for expr in Binome:
    # print(r"%s &= %s\\" % (latex(expr), latex(expand(expr))))
    print(latex(expr), "&=", latex(expand(expr)), r"\\")
print(r"\end{align*}")
```

$$
\begin{aligned}
a + b &= a + b \\
(a + b)^2 &= a^2 + 2ab + b^2 \\
(a + b)^3 &= a^3 + 3a^2b + 3ab^2 + b^3 \\
(a + b)^4 &= a^4 + 4a^3b + 6a^2b^2 + 4ab^3 + b^4 \\
(a + b)^5 &= a^5 + 5a^4b + 10a^3b^2 + 10a^2b^3 + 5ab^4 + b^5 \\
(a + b)^6 &= a^6 + 6a^5b + 15a^4b^2 + 20a^3b^3 + 15a^2b^4 + 6ab^5 + b^6 \\
(a + b)^7 &= a^7 + 7a^6b + 21a^5b^2 + 35a^4b^3 + 35a^3b^4 + 21a^2b^5 + 7ab^6 + b^7 \\
(a + b)^8 &= a^8 + 8a^7b + 28a^6b^2 + 56a^5b^3 + 70a^4b^4 + 56a^3b^5 + 28a^2b^6 + 8ab^7 + b^8 \\
(a + b)^9 &= a^9 + 9a^8b + 36a^7b^2 + 84a^6b^3 + 126a^5b^4 + 126a^4b^5 + 84a^3b^6 + 36a^2b^7 + 9ab^8 + b^9
\end{aligned}
$$

```latex
% pytex12.tex
% aus Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% pyblock, \printpythontex, Modul sympy, Binome, Schleife, Tabelle
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}      % Mathematik à la AMS, hier für align*

\begin{document}
\section*{PythonTeX: pyblock, printpythontex, sympy, Binome, Schleife, Tabelle}

\begin{pyblock}
from sympy import *           # symbolische Mathematik
var("a, b")                   # sympy-Variable
Binome = []                   # Liste für Binomi-Ausdrücke vorbesetzt

for m in range(1, 10):
    Binome.append((a + b)**m) # Binomi-Ausdrücke erzeugen

# Aufbau einer Tabelle mit einer align*-Umgebung
print(r"\begin{align*}")
for expr in Binome:
    # print(r"%s &= %s\\" % (latex(expr), latex(expand(expr))))
    print(latex(expr), "&=", latex(expand(expr)), r"\\")
print(r"\end{align*}")
\end{pyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python12.py
# nach Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{amsmath}      % Mathematik à la AMS, hier für align*")

print(r"\begin{document}")
print(r"\section*{python12: sympy, Binome, Schleife, Tabelle}")

from sympy import *            # symbolische Mathematik
var("a, b")                    # sympy-Variable
Binome = []                    # Liste für Binomi-Ausdrücke vorbesetzt

for m in range(1, 10):
    Binome.append((a + b)**m) # Binomi-Ausdrücke erzeugen

print(r"\begin{align*}")       # Aufbau einer Tabelle mit einer align*-Umgebung
for expr in Binome:
    print(latex(expr), "&=", latex(expand(expr)), r"\\")
print(r"\end{align*}")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{amsmath}      % Mathematik à la AMS, hier für align*
\begin{document}
\section*{python12: sympy, Binome, Schleife, Tabelle}
\begin{align*}
a + b &= a + b \\
\left(a + b\right)^{2} &= a^{2} + 2 a b + b^{2} \\
\left(a + b\right)^{3} &= a^{3} + 3 a^{2} b + 3 a b^{2} + b^{3} \\
\left(a + b\right)^{4} &= a^{4} + 4 a^{3} b + 6 a^{2} b^{2} + 4 a b^{3} + b^{4} \\
\left(a + b\right)^{5} &= a^{5} + 5 a^{4} b + 10 a^{3} b^{2} + 10 a^{2} b^{3} + 5 a b^{4}
\left(a + b\right)^{6} &= a^{6} + 6 a^{5} b + 15 a^{4} b^{2} + 20 a^{3} b^{3} + 15 a^{2} b
\left(a + b\right)^{7} &= a^{7} + 7 a^{6} b + 21 a^{5} b^{2} + 35 a^{4} b^{3} + 35 a^{3} b
\left(a + b\right)^{8} &= a^{8} + 8 a^{7} b + 28 a^{6} b^{2} + 56 a^{5} b^{3} + 70 a^{4} b
\left(a + b\right)^{9} &= a^{9} + 9 a^{8} b + 36 a^{7} b^{2} + 84 a^{6} b^{3} + 126 a^{5}
\end{align*}
\end{document}
```

PythonTeX: pyblock, printpythontex, sympy, printpythontex, Gleichungssystem

```
import sympy as sy          # symbolische Mathematik
h, z, e = sy.var('h z e')   # Sympy-Variablen initiieren

gls = [(z + h + e - 18),    # Gleichungssystem formulieren
(h - 6 - 2 * z),
(e - 6 - 3 * z)]
```

Gleichungssystem

```
print(r"\begin{align*}")    # Gleichungssystem ausgeben in align*-Umgebung
for f in range(len(gls)):
    print(gls[f], r'&= 0\\')
print(r"\end{align*}")
```

$$\begin{aligned} e + h + z - 18 &= 0 \\ h - 2 * z - 6 &= 0 \\ e - 3 * z - 6 &= 0 \end{aligned}$$

Lösung

```
ergebnis = sy.solve(gls)  # Gleichungssystem lösen
for f in ergebnis:
    print(f, ":", ergebnis[f], r"\\")
```

e : 9
h : 8
z : 1

```latex
% pytex24a
%%% pyblock, \printpythontex, Modul sympy, Gleichungssystem, solve
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}   % Mathematik à la AMS
\usepackage{graphicx}  % Einbinden von Graphiken

\begin{document}

\section*{PythonTeX: pyblock, printpythontex, sympy, printpythontex, Gleichungssystem}

\begin{pyblock}
import sympy as sy         # symbolische Mathematik
h, z, e = sy.var('h z e')  # Sympy-Variablen initiieren

gls = [(z + h + e - 18),   # Gleichungssystem formulieren
(h - 6 - 2 * z),
(e - 6 - 3 * z)]
\end{pyblock}

\subsection*{Gleichungssystem}

\begin{pyblock}
print(r"\begin{align*}")   # Gleichungssystem ausgeben in align*-Umgebung
for f in range(len(gls)):
    print(gls[f], r'&= 0\\')
print(r"\end{align*}")
\end{pyblock}
\printpythontex

\subsection*{Lösung}

\begin{pyblock}
ergebnis = sy.solve(gls)  # Gleichungssystem lösen
for f in ergebnis:
    print(f, ":", ergebnis[f], r"\\")
\end{pyblock}
\printpythontex
\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python24a.py
# Modul sympy, Gleichungssystem, solve
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{amsmath}   % Mathematik à la AMS")
print(r"\usepackage{graphicx}  % Einbinden von Graphiken")

print(r"\begin{document}")

print(r"\section*{Python24a: sympy, Gleichungssystem}")

import sympy as sy          # symbolische Mathematik
h, z, e = sy.var('h z e')   # Sympy-Variablen initiieren

gls = [(z + h + e - 18),    # Gleichungssystem formulieren
(h - 6 - 2 * z),
(e - 6 - 3 * z)]

print(r"\subsection*{Gleichungssystem}")

print(r"\begin{align*}")    # Gleichungssystem ausgeben in align*-Umgebung
for f in range(len(gls)):
    print(gls[f], r'&= 0\\')
print(r"\end{align*}")

print(r"\subsection*{Loesung}")

ergebnis = sy.solve(gls)  # Gleichungssystem lösen
for f in ergebnis:
    print(f, ":", ergebnis[f], r"\\")
print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{amsmath}   % Mathematik à la AMS
\usepackage{graphicx}  % Einbinden von Graphiken
\begin{document}
\section*{Python24a: sympy, Gleichungssystem}
\subsection*{Gleichungssystem}
\begin{align*}
e + h + z - 18 &= 0\\
h - 2*z - 6 &= 0\\
e - 3*z - 6 &= 0\\
\end{align*}
\subsection*{Loesung}
e : 9 \\
h : 8 \\
z : 1 \\
\end{document}
```

Python52: Modul scipy.stats, Korrelationkoeffizienten

```
from scipy.stats import spearmanr, pearsonr

name  = "rohdaten2.csv"; X = []; Y = []
daten = open(name, "r")
for zeile in daten:
    einezeile = zeile.rstrip(); werte = einezeile.split(",")
    X = X + [float(werte[0])]; Y = Y + [float(werte[1])]
daten.close()
sc, sp = spearmanr(X, Y); pc, pp = pearsonr(X, Y)

l2 = len(X) // 2
print(r"\begin{tabular}[t]{r|ll}")
print("$i$ & $X_i$ & $Y_i$", r"\\ \hline")
for f in range(l2):
    print(f, "&", X[f], "&", Y[f], r"\\")
print(r"\end{tabular}")

print(r"\begin{tabular}[t]{r|ll}")
print("$i$ & $X_i$ & $Y_i$", r"\\ \hline")
for f in range(l2, len(X)):
    print(f, "&", X[f], "&", Y[f], r"\\")
print(r"\end{tabular}")
```

i	X_i	Y_i	i	X_i	Y_i
0	32.50234526945303	31.70700584656992	48	33.86421497177824	52.725494375900425
1	53.42680403327502	68.77759598163891	49	64.7071386661213	93.57611869265824
2	61.53035802563644	62.5623822979458	50	38.119824026822805	80.16627544737096
3	47.4756396347861	71.54663223356778	51	44.5025380646451	65.10171157056033
4	59.81320786951232	87.23092513368739	52	40.59953838455232	65.56230126040037
5	55.14218841394382	78.21151827079923	53	41.72067635634129	65.28088692082282
6	52.211796692214	79.64197304980874	54	51.088634678336796	73.4346415463243
7	39.299566694317065	59.17148932186951	55	55.0780959049232	71.13972785861894
8	48.10504169176825	75.33124229706306	56	41.3777265348952	79.10282968354986
9	52.55001444273382	71.30087988685035	57	62.49469742726979	86.52053844034715
10	45.419730144973755	55.16567714595912	58	49.203887540826	84.74269780782622
11	54.35163488122892	82.47884675749792	59	41.102685187349664	59.35885024862493
12	44.16404949677335	62.008923245725825	60	41.18201610516982	61.68403752483363
13	58.16847071685779	75.39287042599496	61	50.1863894948806	69.84760415824918
14	56.72720805709661	81.43619215887864	62	52.37844621923622	86.0982912057741
15	48.95588856609372	60.723602440673965	63	50.13548548628612	59.10883926769964
16	44.687196231480904	82.89250373145371	64	33.64470600619178	69.89968164362763
17	60.297326851333466	97.37989686216608	65	39.55790122290683	44.8624907111644
18	45.61864377295583	48.84715331735507	66	56.13038881687547	85.49806777884022
19	38.81681753744564	56.877213186268506	67	57.36205213323824	95.53668684646722
20	66.1898166067526	83.87856466460276	68	60.269214393997906	70.25193441977159
21	65.41605174513407	118.5912173025225	69	35.67809388941073	52.72173496477499
22	47.48120860786787	57.25181946226897	70	31.58811699813283	50.3926701350799
23	41.57564261748702	51.39174407983231	71	53.66093226167304	63.64239877565775
24	51.84518690563943	75.38065166531236	72	46.68222864947192	72.24725106866237
25	59.37082201108952	74.76556403215137	73	43.107820219102464	57.8125129761814
26	57.31000343834809	95.45505292257474	74	70.34607561504933	104.25710158543822
27	63.61556125145331	95.22936601755531	75	44.49285588085407	86.642020318822
28	46.73761940797697	79.05240616956559	76	57.50453330326841	91.48677800011014
29	50.55676014854777	83.43207142132371	77	36.93007660919181	55.231660886212836
30	52.22399608555305	63.35879031749788	78	55.80573335794274	79.55043667850761
31	35.56783004774663	41.41288530370056	79	38.954769073377065	44.8471242424676
32	42.43647694405564	76.61734128007404	80	56.901214702247074	80.20752313968276
33	58.16454011019286	96.7695664261082	81	56.868900661384046	83.14274979204346
34	57.50444761534179	74.08413011660252	82	34.33312470421609	55.723489260543914
35	45.44053072531998	66.5881444142286	83	59.04974121466681	77.63418251167786
36	61.89622268029126	77.76848241779302	84	57.78822399323067	99.05141484174827
37	33.09383173616396	50.71958891231208	85	54.28232870596741	79.12064627468003
38	36.43600951138687	62.12457081807178	86	51.08871989897914	69.58889785111847
39	37.67565486085074	60.81024664990221	87	50.28283634823073	69.51050331149439
40	44.555608383275356	52.68298336638778	88	44.21174175209011	73.68756431831729
41	43.31828263186572	58.56982471769287	89	38.00548800806069	61.36690453724013
42	50.073145632289034	82.90598148507051	90	32.940479942618296	67.17065576899512
43	43.87061264521837	61.42470980433912	91	53.691639571070056	85.66820314500154
44	62.99748074755309	115.24415280079529	92	68.76573426962166	114.85387123391394
45	32.66904376346719	45.570588823376085	93	46.23096649831025	90.12357206996742
46	40.1668990087037	54.08405479622361	94	68.31936081825536	97.91982103524285
47	53.57507753167366	87.99445275811041	95	50.03017434031214	81.53699078301503

Anzahl der Wertepaare	96
Pearson-Korrelationskoeffizient	0.7757599935211379
Spearman-Korrelationskoeffizient	0.7591562669560499

```latex
% pytex52.tex
% nach https://kite.com/python/examples/665/scipy-compute-the-spearman-correlation
% + https://kite.com/python/examples/656/scipy-compute-the-pearson-correlation-coefficient
% Modul scipy.stats, Korrelationkoeffizienten nach Pearson und Spearman
\documentclass[parskip=half,fontsize=10,DIV=9,paper=a4]{scrartcl}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{pythontex}
\usepackage{lmodern}
\usepackage{graphicx}

\begin{document}
\section*{Python52: Modul scipy.stats, Korrelationkoeffizienten}

\begin{pyblock}
from scipy.stats import spearmanr, pearsonr

name  = "rohdaten2.csv"; X = []; Y = []
daten = open(name, "r")
for zeile in daten:
    einezeile = zeile.rstrip(); werte = einezeile.split(",")
    X = X + [float(werte[0])]; Y = Y + [float(werte[1])]
daten.close()
sc, sp = spearmanr(X, Y); pc, pp = pearsonr(X, Y)

l2 = len(X) // 2
print(r"\begin{tabular}[t]{r|ll}")
print("$i$ & $X_i$ & $Y_i$", r"\\ \hline")
for f in range(l2):
    print(f, "&", X[f], "&", Y[f], r"\\")
print(r"\end{tabular}")

print(r"\begin{tabular}[t]{r|ll}")
print("$i$ & $X_i$ & $Y_i$", r"\\ \hline")
for f in range(l2, len(X)):
    print(f, "&", X[f], "&", Y[f], r"\\")
print(r"\end{tabular}")
\end{pyblock}
\printpythontex

%print(r"\begin{tabular}[t]{r|llr|ll}")
%print("$i$ & $X_i$ & $Y_i$ & $i$ & $X_i$ & $Y_i$", r"\\ \hline")
%for f in range(l2):
    %print(f+1, "&", X[f], "&", Y[f], "&", f+l2, "&", X[f+l2], "&", Y[f+l2], r"\\")
```

```
%print(r"\end{tabular}")

\begin{tabular}{ll}
Anzahl der Wertepaare            & \py{len(X)}  \\
Pearson-Korrelationskoeffizient  & \py{pc} \\
Spearman-Korrelationskoeffizient & \py{sc} \\
\end{tabular}

\end{document}
```

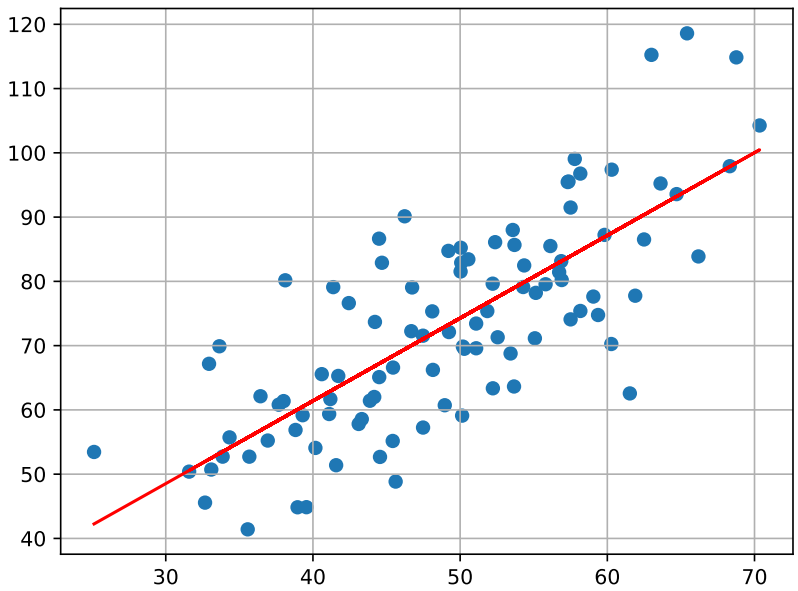
```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python52.py
# nach https://kite.com/python/examples/665/scipy-compute-the-spearman-correlation
# + https://kite.com/python/examples/656/scipy-compute-the-pearson-correlation-coefficient
# Modul scipy.stats, Korrelationkoeffizienten nach Pearson und Spearman
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{graphicx}")

print(r"\begin{document}")
print(r"\section*{Python52: Modul scipy.stats, Korrelationkoeffizienten}")

from scipy.stats import spearmanr, pearsonr
name = "rohdaten2.csv"; X = []; Y = []
daten = open(name, "r")
for zeile in daten:
    einezeile = zeile.rstrip(); werte = einezeile.split(",")
    X = X + [float(werte[0])]; Y = Y + [float(werte[1])]
daten.close()
sc, sp = spearmanr(X, Y); pc, pp = pearsonr(X, Y)

l2 = len(X) // 2

print(r"\begin{tabular}[t]{r|ll}")
print("$i$ & $X_i$ & $Y_i$", r"\\ \hline")
for f in range(l2):
    print(f, "&", X[f], "&", Y[f], r"\\")
print(r"\end{tabular}")

print(r"\begin{tabular}[t]{r|ll}")
print("$i$ & $X_i$ & $Y_i$", r"\\ \hline")
for f in range(l2, len(X)):
    print(f, "&", X[f], "&", Y[f], r"\\")
print(r"\end{tabular}")

print(r"\begin{tabular}{ll}")
print("Anzahl der Wertepaare &", len(X), r"\\")
print("Pearson-Korrelationskoeffizient &", pc, r"\\")
print("Spearman-Korrelationskoeffizient &", sc, r"\\")
print(r"\end{tabular}")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{graphicx}
\begin{document}
\section*{Python52: Modul scipy.stats, Korrelationkoeffizienten}
\begin{tabular}[t]{r|llr|ll}
$i$ & $X_i$ & $Y_i$ & $i$ & $X_i$ & $Y_i$ \\ \hline
1 & 32.50234526945303 & 31.70700584656992 & 48 & 33.86421497177824 & 52.725494375900425 \\
2 & 53.42680403327502 & 68.77759598163891 & 49 & 64.7071386661213 & 93.57611869265824 \\
3 & 61.53035802563644 & 62.5623822979458 & 50 & 38.119824026822805 & 80.16627544737096 \\
4 & 47.4756396347861 & 71.54663223356778 & 51 & 44.5025380646451 & 65.10171157056033 \\
5 & 59.81320786951232 & 87.23092513368739 & 52 & 40.59953838455232 & 65.56230126040037 \\
6 & 55.14218841394382 & 78.21151827079923 & 53 & 41.72067635634129 & 65.28088692082282 \\
7 & 52.211796692214 & 79.64197304980874 & 54 & 51.088634678336796 & 73.4346415463243 \\
8 & 39.299566694317065 & 59.17148932186951 & 55 & 55.0780959049232 & 71.13972785861894 \\
9 & 48.10504169176825 & 75.33124229706306 & 56 & 41.3777265348952 & 79.10282968354986 \\
10 & 52.55001444273382 & 71.30087988685035 & 57 & 62.49469742726979 & 86.52053844034715 \\
11 & 45.419730144973755 & 55.16567714595912 & 58 & 49.203887540826 & 84.74269780782622 \\
12 & 54.35163488122892 & 82.47884675749792 & 59 & 41.102685187349664 & 59.35885024862493 \
13 & 44.16404949677335 & 62.008923245725825 & 60 & 41.18201610516982 & 61.68403752483363 \
14 & 58.16847071685779 & 75.39287042599496 & 61 & 50.1863894948806 & 69.84760415824918 \\
15 & 56.72720805709661 & 81.43619215887864 & 62 & 52.37844621923622 & 86.0982912057741 \\
16 & 48.95588856609372 & 60.723602440673965 & 63 & 50.13548548628612 & 59.10883926769964 \
17 & 44.687196231480904 & 82.89250373145371 & 64 & 33.64470600619178 & 69.89968164362763 \
18 & 60.297326851333466 & 97.37989686216608 & 65 & 39.55790122290683 & 44.8624907111644 \\
19 & 45.61864377295583 & 48.84715331735507 & 66 & 56.13038881687547 & 85.49806777884022 \\
20 & 38.81681753744564 & 56.877213186268506 & 67 & 57.36205213323824 & 95.53668684646722 \
21 & 66.1898166067526 & 83.87856466460276 & 68 & 60.269214393997906 & 70.25193441977159 \\
22 & 65.41605174513407 & 118.5912173025225 & 69 & 35.67809388941073 & 52.72173496477499 \\
23 & 47.48120860786787 & 57.25181946226897 & 70 & 31.58811699813283 & 50.3926701350799 \\
24 & 41.57564261748702 & 51.39174407983231 & 71 & 53.66093226167304 & 63.64239877565775 \\
25 & 51.84518690563943 & 75.38065166531236 & 72 & 46.68222864947192 & 72.24725106866237 \\
26 & 59.37082201108952 & 74.76556403215137 & 73 & 43.107820219102464 & 57.8125129761814 \\
27 & 57.31000343834809 & 95.45505292257474 & 74 & 70.34607561504933 & 104.25710158543822 \
28 & 63.61556125145331 & 95.22936601755531 & 75 & 44.49285588085407 & 86.642020318822 \\
29 & 46.73761940797697 & 79.05240616956559 & 76 & 57.50453330326841 & 91.48677800011014 \\
30 & 50.55676014854777 & 83.43207142132371 & 77 & 36.93007660919181 & 55.231660886212836 \
31 & 52.22399608555305 & 63.35879031749788 & 78 & 55.80573335794274 & 79.55043667850761 \\
32 & 35.56783004774663 & 41.41288530370056 & 79 & 38.954769073377065 & 44.8471242424676 \\
33 & 42.43647694405564 & 76.61734128007404 & 80 & 56.901214702247074 & 80.20752313968276 \
34 & 58.16454011019286 & 96.7695664261082 & 81 & 56.868900661384046 & 83.14274979204346 \\
35 & 57.50444761534179 & 74.08413011660252 & 82 & 34.33312470421609 & 55.723489260543914 \
36 & 45.44053072531998 & 66.5881444142286 & 83 & 59.04974121466681 & 77.63418251167786 \\
37 & 61.89622268029126 & 77.76848241779302 & 84 & 57.78822399323067 & 99.05141484174827 \\
```

```latex
38 & 33.09383173616396 & 50.71958891231208 & 85 & 54.28232870596741 & 79.12064627468003 \\
39 & 36.43600951138687 & 62.12457081807178 & 86 & 51.08871989897914 & 69.58889785111847 \\
40 & 37.67565486085074 & 60.81024664990221 & 87 & 50.28283634823073 & 69.51050331149439 \\
41 & 44.555608383275356 & 52.68298336638778 & 88 & 44.21174175209011 & 73.68756431831729 \
42 & 43.31828263186572 & 58.56982471769287 & 89 & 38.00548800806069 & 61.36690453724013 \\
43 & 50.073145632289034 & 82.90598148507051 & 90 & 32.940479942618296 & 67.17065576899512
44 & 43.87061264521837 & 61.42470980433912 & 91 & 53.691639571070056 & 85.66820314500154 \
45 & 62.99748074755309 & 115.24415280079529 & 92 & 68.76573426962166 & 114.85387123391394
46 & 32.66904376346719 & 45.570588823376085 & 93 & 46.23096649831025 & 90.12357206996742 \
47 & 40.1668990087037 & 54.08405479622361 & 94 & 68.31936081825536 & 97.91982103524285 \\
48 & 53.57507753167366 & 87.99445275811041 & 95 & 50.03017434031214 & 81.53699078301503 \\
\end{tabular}
\begin{tabular}{ll}
Anzahl der Wertepaare & 96 \\
Pearson-Korrelationskoeffizient & 0.7757599935211379 \\
Spearman-Korrelationskoeffizient & 0.7591562669560499 \\
\end{tabular}
\end{document}
```

Python51: numpy, Grafik mit matplotlib, sklearn, Regression

```
import matplotlib.pyplot as plt            # visualisieren
import pandas as pd                        # Daten einlesen
from sklearn.linear_model import LinearRegression

daten= pd.read_csv('rohdaten.csv')         # Daten einlesen
X    = daten.iloc[:, 0].values.reshape(-1, 1)  # in numpy-Liste umwandeln
Y    = daten.iloc[:, 1].values.reshape(-1, 1)  # + Abmessungen berechnen
linear_regressor = LinearRegression()      # Objekt generieren
linear_regressor.fit(X, Y)                 # lineare Regression
Y_pred = linear_regressor.predict(X)       # Werte der Regressionsgeraden
plt.scatter(X, Y)                          # Punkte zeichnen
plt.plot(X, Y_pred, color='red')           # Regressionsgerade zeichnen
plt.grid(True)                             # Gitter zeichnen
plt.savefig("regression.pdf")              # als PDF abspeichern

print(r"\begin{center}")
print(r"\includegraphics[width=0.85\textwidth]{regression.pdf}")
print(r"\end{center}")
```



```latex
% pytex51.tex
% aus https://github.com/chasinginfinity/ml-from-scratch/blob/master/03 Linea Regression i
% numpy, Grafik mit matplotlib, sklearn, Regression
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{graphicx}

\begin{document}
\section*{Python51: numpy, Grafik mit matplotlib, sklearn, Regression}

\begin{pyblock}
import matplotlib.pyplot as plt            # visualisieren
import pandas as pd                        # Daten einlesen
from sklearn.linear_model import LinearRegression

daten= pd.read_csv('rohdaten.csv')         # Daten einlesen
X    = daten.iloc[:, 0].values.reshape(-1, 1)  # in numpy-Liste umwandeln
Y    = daten.iloc[:, 1].values.reshape(-1, 1)  # + Abmessungen berechnen
linear_regressor = LinearRegression()      # Objekt generieren
linear_regressor.fit(X, Y)                 # lineare Regression
Y_pred = linear_regressor.predict(X)       # Werte der Regressionsgeraden
plt.scatter(X, Y)                          # Punkte zeichnen
plt.plot(X, Y_pred, color='red')           # Regressionsgerade zeichnen
plt.grid(True)                             # Gitter zeichnen
plt.savefig("regression.pdf")              # als PDF abspeichern

print(r"\begin{center}")
print(r"\includegraphics[width=0.85\textwidth]{regression.pdf}")
print(r"\end{center}")
\end{pyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python51.py
# aus https://github.com/chasinginfinity/ml-from-scratch/blob/master/03%20Linear%20Regress
# numpy, Grafik mit matplotlib, sklearn, Regression
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{graphicx}")

print(r"\begin{document}")
print(r"\section*{Python51: numpy, Grafik mit matplotlib, sklearn, Regression}")
import matplotlib.pyplot as plt            # visualisieren
import pandas as pd                        # Daten einlesen
from sklearn.linear_model import LinearRegression

daten= pd.read_csv('rohdaten.csv')         # Daten einlesen
X    = daten.iloc[:, 0].values.reshape(-1, 1)  # in numpy-Liste umwandeln
Y    = daten.iloc[:, 1].values.reshape(-1, 1)  # + Abmessungen berechnen
linear_regressor = LinearRegression()      # Objekt generieren
linear_regressor.fit(X, Y)                 # lineare Regression
Y_pred = linear_regressor.predict(X)       # Predictions
plt.scatter(X, Y)                          # Punkte zeichnen
plt.plot(X, Y_pred, color='red')           # Regressionsgerade zeichnen
plt.grid(True)                             # Gitter zeichnen
plt.savefig("regression.pdf")              # als PDF abspeichern

print(r"\begin{center}")
print(r"\includegraphics[width=0.85\textwidth]{regression.pdf}")
print(r"\end{center}")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{graphicx}
\begin{document}
\section*{Python51: numpy, Grafik mit matplotlib, sklearn, Regression}
\begin{center}
\includegraphics[width=0.85\textwidth]{regression.pdf}
\end{center}
\end{document}
```

PythonTeX: sympy, pyblock, printpythontex, Funktionen, Integrale, Tabelle

```python
from sympy import *                                    # symbolische Mathematik
var("x")                                               # symbolische Variable

funktionen = [sin(x), cos(x), tan(x), cot(x)]          # zu betrachtende Funktionen

print(r"\begin{align*}")                               # ATabelle mit align*
for f in funktionen:
    i = Integral(f, x)                                 # Integral: Integral, formal
    print(latex(i) + "&=" + latex(i.doit()) + r"\\") # .doit():  ausgewertet
print(r"\end{align*}")
```

$$\begin{aligned}
\int \sin(x)\,dx &= -\cos(x) \\
\int \cos(x)\,dx &= \sin(x) \\
\int \tan(x)\,dx &= -\frac{1}{2}\log\left(\sin^2(x) - 1\right) \\
\int \cot(x)\,dx &= \frac{1}{2}\log\left(\cos^2(x) - 1\right)
\end{aligned}$$

```latex
% pytex13
% nach Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% Modul sympy, pyblock, \printpythontex, Funktionen, Integral, doit, latex, Tabelle,
\documentclass[parskip=half, fontsize=11, paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}     % Mathematik à la AMS

\begin{document}
\section*{PythonTeX: sympy, pyblock, printpythontex, Funktionen, Integrale, Tabelle}

\begin{pyblock}
from sympy import *                                  # symbolische Mathematik
var("x")                                             # symbolische Variable

funktionen = [sin(x), cos(x), tan(x), cot(x)]        # zu betrachtende Funktionen

print(r"\begin{align*}")                             # ATabelle mit align*
for f in funktionen:
    i = Integral(f, x)                               # Integral: Integral, formal
    print(latex(i) + "&=" + latex(i.doit()) + r"\\") # .doit():  ausgewertet
print(r"\end{align*}")
\end{pyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python13.py
# nach Mertz, Slough 2013 - A Gentle Introduction to PythonTeX

print(r"\documentclass[parskip=half, fontsize=11, paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{amsmath}     % Mathematik à la AMS")

print(r"\begin{document}")
print(r"\section*{python13: sympy, Funktionen, Integrale, Tabelle}")

##from math import *                                    # mathematische Funktionen
from sympy import *                                   # symbolische Mathematik
var("x")                                              # symbolische Variable

funktionen = [sin(x), cos(x), tan(x), cot(x)]         # zu betrachtende Funktionen

print(r"\begin{align*}")                              # Tabelle mit align*
for f in funktionen:
    i = Integral(f, x)                                # Integral: Integral, formal
    print(latex(i) + "&=" + latex(i.doit()) + r"\\") # .doit():  ausgewertet
print(r"\end{align*}")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half, fontsize=11, paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{amsmath}     % Mathematik à la AMS
\begin{document}
\section*{python13: sympy, math, Funktionen, Integrale, Tabelle}
\begin{align*}
\int \sin{\left (x \right )}\, dx&=- \cos{\left (x \right )}\\
\int \cos{\left (x \right )}\, dx&=\sin{\left (x \right )}\\
\int \tan{\left (x \right )}\, dx&=- \frac{1}{2} \log{\left (\sin^{2}{\left (x \right )} -
\int \cot{\left (x \right )}\, dx&=\frac{1}{2} \log{\left (\cos^{2}{\left (x \right )} - 1
\end{align*}
\end{document}
```

PythonTeX: Modul sympy, pyblock, printpythontex, unbestimmtes und bestimmtes Integral, Tabelle

```
#from sympy import *                          # smbolische Mathematik
var('x')                                     # symbolische Variable
phi = Symbol(r'\phi')                        # Symbol phi
pi  = Symbol(r'\pi')                         # Symbol pi
f   = x**3 + cos(x)**5                       # zu untersuchende erste Funktion
g   = Integral(f, x)                         # unbestimmtes Integral dazu
gi  = Integral(f, (x, 0, pi/2))              # bestimmtes Integral dazu
hi  = Integral(exp(-phi**2), (phi, 0, oo)) # bestimmtes Integral für 2. Funktion

# align*-Umgebung wird erstellt
print(r"\begin{align*}")                     # Tabelle; LaTeX-Ausgabe mittels latex()
print(latex(g)  + "&=" + latex(g.doit()) + r"\\")
print(latex(gi) + "&=" + latex(gi.doit()) + r"\\")
print(            "&=" + latex(N(gi.doit())) + r"\\")
print(latex(hi) + "&=" + latex(hi.doit()) + r"\\")
print(            "&=" + latex(N(hi.doit())) + r"\\")
print(r"\end{align*}")
```

$$\begin{aligned}
\int x^3 + \cos^5(x)\,dx &= \frac{x^4}{4} + \frac{1}{5}\sin^5(x) - \frac{2}{3}\sin^3(x) + \sin(x) \\
\int_0^{\frac{\pi}{2}} x^3 + \cos^5(x)\,dx &= \frac{\pi^4}{64} + \frac{1}{5}\sin^5\left(\frac{\pi}{2}\right) - \frac{2}{3}\sin^3\left(\frac{\pi}{2}\right) + \sin\left(\frac{\pi}{2}\right) \\
&= 0.015625\pi^4 + 0.2\sin^5\left(\frac{\pi}{2}\right) - 0.666666666666667\sin^3\left(\frac{\pi}{2}\right) + \sin\left(\frac{\pi}{2}\right) \\
\int_0^{\infty} e^{-\phi^2}\,d\phi &= \frac{\sqrt{\pi}}{2} \\
&= 0.886226925452758
\end{aligned}$$

```latex
% pytex14
%%% Modul sympy, pyblock, \printpythontex, unbestimmtes und bestimmtes Integral, doit, lat
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[intlimits]{amsmath}        % Mathematik à la AMS
\begin{document}

\section*{PythonTeX: Modul sympy, pyblock, printpythontex, unbestimmtes und bestimmtes Int

\begin{sympyblock}
#from sympy import *                        # smbolische Mathematik
var('x')                                   # symbolische Variable
phi = Symbol(r'\phi')                      # Symbol phi
pi  = Symbol(r'\pi')                       # Symbol pi
f   = x**3 + cos(x)**5                     # zu untersuchende erste Funktion
g   = Integral(f, x)                       # unbestimmtes Integral dazu
gi  = Integral(f, (x, 0, pi/2))            # bestimmtes Integral dazu
hi  = Integral(exp(-phi**2), (phi, 0, oo)) # bestimmtes Integral für 2. Funktion

# align*-Umgebung wird erstellt
print(r"\begin{align*}")                   # Tabelle; LaTeX-Ausgabe mittels latex()
print(latex(g)  + "&=" + latex(g.doit()) + r"\\")
print(latex(gi) + "&=" + latex(gi.doit()) + r"\\")
print(            "&=" + latex(N(gi.doit())) + r"\\")
print(latex(hi) + "&=" + latex(hi.doit()) + r"\\")
print(            "&=" + latex(N(hi.doit())) + r"\\")
print(r"\end{align*}")
\end{sympyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python14.py
# Modul sympy, unbestimmtes und bestimmtes Integral, doit, latex, Tabelle
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage[intlimits]{amsmath}        % Mathematik à la AMS")
print(r"\begin{document}")

print(r"\section*{Python14: Modul sympy, unbestimmtes und bestimmtes Integral, Tabelle}")

from sympy import *                          # smbolische Mathematik
var('x')                                     # symbolische Variable
phi = Symbol(r'\phi')                        # Symbol phi
pi  = Symbol(r'\pi')                         # Symbol pi
f   = x**3 + cos(x)**5                       # zu untersuchende erste Funktion
g   = Integral(f, x)                         # unbestimmtes Integral dazu
gi  = Integral(f, (x, 0, pi/2))              # bestimmtes Integral dazu
hi  = Integral(exp(-phi**2), (phi, 0, oo)) # bestimmtes Integral für 2. Funktion

# align*-Umgebung wird erstellt
print(r"\begin{align*}")                     # Tabelle; LaTeX-Ausgabe mittels latex()
print(latex(g)  + "&=" + latex(g.doit()) + r"\\")
print(latex(gi) + "&=" + latex(gi.doit()) + r"\\")
print(            "&=" + latex(N(gi.doit())) + r"\\")
print(latex(hi) + "&=" + latex(hi.doit()) + r"\\")
print(            "&=" + latex(N(hi.doit())) + r"\\")
print(r"\end{align*}")

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage[intlimits]{amsmath}        % Mathematik à la AMS
\begin{document}
\section*{Python14: Modul sympy, unbestimmtes und bestimmtes Integral, Tabelle}
\begin{align*}
\int x^{3} + \cos^{5}{\left (x \right )}\, dx&=\frac{x^{4}}{4} + \frac{1}{5} \sin^{5}{\lef
\int_{0}^{\frac{\pi}{2}} x^{3} + \cos^{5}{\left (x \right )}\, dx&=\frac{\pi^{4}}{64} + \f
\int_{0}^{\infty} e^{- \phi^{2}}\, d\phi&=\frac{\sqrt{\pi}}{2}\\
\end{align*}
\end{document}
```

PythonTeX: pyblock, printpythontex, sympy, bestimmtes Mehrfach-Integral

```python
from sympy import *             # symbolische Mathematik
x, y, z = symbols('x, y, z')    # sympy-Variable
f       = Symbol('f(x, y, z)') # sympy-Variable

# Integrationsgrenzen
x_ug  = 0
x_og  = 2
y_ug  = 0
y_og  = 3
z_ug  = 0
z_og  = 4

print(r'\begin{align*}')
links  = Integral(f, (x, x_ug, x_og), (y, y_ug, y_og), (z, z_ug, z_og))

f      = x * y + y * sin(z) + cos(x + y)
rechts = Integral(f, (x, x_ug, x_og), (y, y_ug, y_og), (z, z_ug, z_og))
print(latex(links) + '&=' + latex(rechts) + r'\\')

# Schritt 1: Integral mit bestimmten Grenzen für z
rechts = Integral(Integral(f, (z, z_ug, z_og)).doit(), (x, x_ug, x_og),
         (y, y_ug, y_og))
print('&=' + latex(rechts) + r'\\')

# Schritt 2: Integral mit bestimmten Grenzen für y
rechts = Integral(Integral(f, (z, z_ug, z_og), (y, y_ug, y_og)).doit(),
         (x, x_ug, x_og))
print('&=' + latex(rechts) + r'\\')

# Schritt 3: Integral mit bestimmten Grenzen für x (Gesamtintegral)
rechts = Integral(f, (z, z_ug, z_og), (y, y_ug, y_og),
         (x, x_ug, x_og)).doit()
print('&=' + latex(rechts) + r'\\')

print('&=' + latex(N(rechts)) + r'\\')
print(r'\end{align*}')
```

$$
\begin{aligned}
\int_0^4 \int_0^3 \int_0^2 f(x,y,z)\,dx\,dy\,dz &= \int_0^4 \int_0^3 \int_0^2 xy + y\sin(z) + \cos(x+y)\,dx\,dy\,dz \\
&= \int_0^3 \int_0^2 4xy - y\cos(4) + y + 4\cos(x+y)\,dx\,dy \\
&= \int_0^2 18x - 4\sin(x) + 4\sin(x+3) - \frac{9}{2}\cos(4) + \frac{9}{2}\,dx \\
&= 4\cos(3) + 4\cos(2) - 4\cos(5) - 9\cos(4) + 41 \\
&= 40.1235865133293
\end{aligned}
$$

```latex
% pytex16
% Poore 2017 - PythonTeX Gallery
%%% pyblock, \printpythontex, Modul sympy, bestimmtes Mehrfach-Integral, latex, doit
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[intlimits]{amsmath}   % Mathematik à la AMS

\begin{document}
\section*{PythonTeX: pyblock, printpythontex, sympy, bestimmtes Mehrfach-Integral}

\begin{pyblock}
from sympy import *            # symbolische Mathematik
x, y, z = symbols('x, y, z')   # sympy-Variable
f       = Symbol('f(x, y, z)') # sympy-Variable

# Integrationsgrenzen
x_ug  = 0
x_og  = 2
y_ug  = 0
y_og  = 3
z_ug  = 0
z_og  = 4

print(r'\begin{align*}')
links  = Integral(f, (x, x_ug, x_og), (y, y_ug, y_og), (z, z_ug, z_og))

f      = x * y + y * sin(z) + cos(x + y)
rechts = Integral(f, (x, x_ug, x_og), (y, y_ug, y_og), (z, z_ug, z_og))
print(latex(links) + '&=' + latex(rechts) + r'\\')

# Schritt 1: Integral mit bestimmten Grenzen für z
rechts = Integral(Integral(f, (z, z_ug, z_og)).doit(), (x, x_ug, x_og),
         (y, y_ug, y_og))
print('&=' + latex(rechts) + r'\\')

# Schritt 2: Integral mit bestimmten Grenzen für y
rechts = Integral(Integral(f, (z, z_ug, z_og), (y, y_ug, y_og)).doit(),
         (x, x_ug, x_og))
print('&=' + latex(rechts) + r'\\')

# Schritt 3: Integral mit bestimmten Grenzen für x (Gesamtintegral)
rechts = Integral(f, (z, z_ug, z_og), (y, y_ug, y_og),
```

```
        (x, x_ug, x_og)).doit()
print('&=' + latex(rechts) + r'\\')

print('&=' + latex(N(rechts)) + r'\\')
print(r'\end{align*}')
\end{pyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python16.tex
# Poore 2017 - PythonTeX Gallery
# Modul sympy, bestimmtes Mehrfach-Integral, latex, doit
print(r"\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage[intlimits]{amsmath}   % Mathematik à la AMS")

print(r"\begin{document}")
print(r"\section*{Python16: sympy, bestimmtes Mehrfach-Integral}")

from sympy import *            # symbolische Mathematik
x, y, z = symbols('x, y, z')   # sympy-Variable
f       = Symbol('f(x, y, z)') # sympy-Variable

# Integrationsgrenzen
x_ug  = 0
x_og  = 2
y_ug  = 0
y_og  = 3
z_ug  = 0
z_og  = 4

print(r'\begin{align*}')
links  = Integral(f, (x, x_ug, x_og), (y, y_ug, y_og), (z, z_ug, z_og))

f      = x * y + y * sin(z) + cos(x + y)
rechts = Integral(f, (x, x_ug, x_og), (y, y_ug, y_og), (z, z_ug, z_og))
print(latex(links) + '&=' + latex(rechts) + r'\\')

# Schritt 1: Integral mit bestimmten Grenzen für z
rechts = Integral(Integral(f, (z, z_ug, z_og)).doit(), (x, x_ug, x_og),
         (y, y_ug, y_og))
print('&=' + latex(rechts) + r'\\')

# Schritt 2: Integral mit bestimmten Grenzen für y
rechts = Integral(Integral(f, (z, z_ug, z_og), (y, y_ug, y_og)).doit(),
         (x, x_ug, x_og))
print('&=' + latex(rechts) + r'\\')

# Schritt 3: Integral mit bestimmten Grenzen für x (Gesamtintegral)
rechts = Integral(f, (z, z_ug, z_og), (y, y_ug, y_og),
         (x, x_ug, x_og)).doit()
```

```python
print('&=' + latex(rechts) + r'\\')

print('&=' + latex(N(rechts)) + r'\\')
print(r'\end{align*}')

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=11,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage[intlimits]{amsmath}   % Mathematik à la AMS
\begin{document}
\section*{Python16: sympy, bestimmtes Mehrfach-Integral}
\begin{align*}
\int_{0}^{4}\int_{0}^{3}\int_{0}^{2} f(x, y, z)\, dx\, dy\, dz&=\int_{0}^{4}\int_{0}^{3}\i
&=\int_{0}^{3}\int_{0}^{2} 4 x y - y \cos{\left (4 \right )} + y + 4 \cos{\left (x + y \ri
&=\int_{0}^{2} 18 x - 4 \sin{\left (x \right )} + 4 \sin{\left (x + 3 \right )} - \frac{9}
&=4 \cos{\left (3 \right )} + 4 \cos{\left (2 \right )} - 4 \cos{\left (5 \right )} - 9 \c
&=40.1235865133293\\
\end{align*}
\end{document}
```

PythonTeX: sympy, sympyblock, printpythontex, Ableitung, eq.replace

```
#from sympy import *
x = symbols('x')                                # sympy-Variable

print(r'\begin{align*}')
for funk in [sin(x), sinh(x), csc(x)]:
    links  = Derivative(funk, x)                # Ableitung, formal
    rechts = Derivative(funk, x).doit()         # Ableitung ausführen
    zeile  = latex(links) + '&=' + latex(rechts) + r'\\'
    print(zeile.replace('d', r'\mathrm{d} '))   # d austauschen
print(r'\end{align*}')
```

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}x} \sin\left(x\right) &= \cos\left(x\right) \\
\frac{\mathrm{d}}{\mathrm{d}x} \sinh\left(x\right) &= \cosh\left(x\right) \\
\frac{\mathrm{d}}{\mathrm{d}x} \csc\left(x\right) &= -\cot\left(x\right)\csc\left(x\right)
\end{aligned}
$$

```latex
% pytex43
% Poore 2015 - PythonTeX: Reproducible Documents with PythonTeX
%%% Modul sympy, sympyblock, \printpythontex, Ableitung, eq.replace, latex, Derivative
\documentclass[parskip=half, fontsize=11, paper=a4]{scrartcl}
\usepackage[ngerman]{babel}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}

\begin{document}

\section*{PythonTeX: sympy, sympyblock, printpythontex, Ableitung, eq.replace}

\begin{sympyblock}
#from sympy import *
x = symbols('x')                                # sympy-Variable

print(r'\begin{align*}')
for funk in [sin(x), sinh(x), csc(x)]:
    links  = Derivative(funk, x)                # Ableitung, formal
    rechts = Derivative(funk, x).doit()         # Ableitung ausführen
    zeile  = latex(links) + '&=' + latex(rechts) + r'\\'
    print(zeile.replace('d', r'\mathrm{d} '))  # d austauschen
print(r'\end{align*}')
\end{sympyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python43.py
# Poore 2015 - PythonTeX: Reproducible Documents with PythonTeX
# Modul sympy, Ableitung, eq.replace, latex, Derivative
print(r"\documentclass[parskip=half, fontsize=11, paper=a4]{scrartcl}")
print(r"\usepackage[ngerman]{babel}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{amsmath}")

print(r"\begin{document}")

print(r"\section*{Python43: Ableitung, replace}")

from sympy import *
x = symbols('x')                            # sympy-Variable

print(r'\begin{align*}')
for funk in [sin(x), sinh(x), csc(x)]:
    links  = Derivative(funk, x)               # Ableitung, formal
    rechts = Derivative(funk, x).doit()        # Ableitung ausführen
    zeile  = latex(links) + '&=' + latex(rechts) + r'\\'
    print(zeile.replace('d', r'\mathrm{d} '))  # d austauschen
print(r'\end{align*}')

print(r"\end{document}")
```

```latex
\documentclass[parskip=half, fontsize=11, paper=a4]{scrartcl}
\usepackage[ngerman]{babel}
\usepackage{fontspec}
\usepackage{amsmath}
\begin{document}
\section*{Python43: Ableitung, replace}
\begin{align*}
\frac{\mathrm{d} }{\mathrm{d}  x} \sin{\left (x \right )}&=\cos{\left (x \right )}\\
\frac{\mathrm{d} }{\mathrm{d}  x} \sinh{\left (x \right )}&=\cosh{\left (x \right )}\\
\frac{\mathrm{d} }{\mathrm{d}  x} \csc{\left (x \right )}&=- \cot{\left (x \right )} \csc{
\end{align*}
\end{document}
```

PythonTeX: pyblock, sympy, Funktionen, Ableitungen, Integrale

```
from sympy import *                                # symbolische Mathematik
var('x')                                           # symbolische Variable x

# Liste von Funktionen, die in der Tabelle aufgeführt werden sollen
funktionen = ['acos(x)', 'acosh(x)', 'acot(x)', 'acoth(x)', 'asin(x)', 'asinh(x)',
              'atan(x)', 'atanh(x)', 'cos(x)**2', 'cos(x)', 'cosh(x)', 'cot(x)**2',
              'cot(x)', 'coth(x)', 'erf(x)', 'erfc(x)', 'exp(x)', 'gamma(x)','ln(x)',
              'sin(x)**2', 'sin(x)', 'sinh(x)', 'sqrt(x)', 'tan(x)**2', 'tan(x)',
              'tanh(x)', 'csc(x)']

f          = "f(x) "
fableit    = "f'(x)"
fint       = r"\int\!\!f(x)\,dx"

print(r'\begin{align*}')                           # Tabelle
for funk in funktionen:                            # Schleife
    if funk in ['sin(x)**2']:                      # ggf. neue Seite
        print(r"\displaybreak")
    meineableit = 'Derivative(' + funk + ', x)'    # Ableitung
    meinint     = 'Integral(' + funk + ', x)'      # Integral
    print(latex(f),       r"&=", latex(eval(funk)),                    r'&')
    print(latex(fableit), r"&=", latex(eval(meineableit + '.doit()')), r"&")
    print(latex(fint),    r"&=", latex(eval(meinint + '.doit()')),     r'\\')
print(r'\end{align*}')
```

$f(x) = \operatorname{acos}(x)$	$f'(x) = -\frac{1}{\sqrt{-x^2+1}}$	$\int f(x)\,dx = x\operatorname{acos}(x) - \sqrt{-x^2+1}$
$f(x) = \operatorname{acosh}(x)$	$f'(x) = \frac{1}{\sqrt{x^2-1}}$	$\int f(x)\,dx = x\operatorname{acosh}(x) - \sqrt{x^2-1}$
$f(x) = \operatorname{acot}(x)$	$f'(x) = -\frac{1}{x^2+1}$	$\int f(x)\,dx = x\operatorname{acot}(x) + \frac{1}{2}\log\left(x^2+1\right)$
$f(x) = \operatorname{acoth}(x)$	$f'(x) = \frac{1}{-x^2+1}$	$\int f(x)\,dx = x\operatorname{acoth}(x) + \log(x+1) - \operatorname{acoth}(x)$
$f(x) = \operatorname{asin}(x)$	$f'(x) = \frac{1}{\sqrt{-x^2+1}}$	$\int f(x)\,dx = x\operatorname{asin}(x) + \sqrt{-x^2+1}$
$f(x) = \operatorname{asinh}(x)$	$f'(x) = \frac{1}{\sqrt{x^2+1}}$	$\int f(x)\,dx = x\operatorname{asinh}(x) - \sqrt{x^2+1}$
$f(x) = \operatorname{atan}(x)$	$f'(x) = \frac{1}{x^2+1}$	$\int f(x)\,dx = x\operatorname{atan}(x) - \frac{1}{2}\log\left(x^2+1\right)$
$f(x) = \operatorname{atanh}(x)$	$f'(x) = \frac{1}{-x^2+1}$	$\int f(x)\,dx = x\operatorname{atanh}(x) + \log(x+1) - \operatorname{atanh}(x)$
$f(x) = \cos^2(x)$	$f'(x) = -2\sin(x)\cos(x)$	$\int f(x)\,dx = \frac{x}{2} + \frac{1}{2}\sin(x)\cos(x)$
$f(x) = \cos(x)$	$f'(x) = -\sin(x)$	$\int f(x)\,dx = \sin(x)$
$f(x) = \cosh(x)$	$f'(x) = \sinh(x)$	$\int f(x)\,dx = \sinh(x)$
$f(x) = \cot^2(x)$	$f'(x) = \left(-2\cot^2(x) - 2\right)\cot(x)$	$\int f(x)\,dx = -x - \frac{\cos(x)}{\sin(x)}$
$f(x) = \cot(x)$	$f'(x) = -\cot^2(x) - 1$	$\int f(x)\,dx = \frac{1}{2}\log\left(\cos^2(x) - 1\right)$
$f(x) = \coth(x)$	$f'(x) = -\frac{1}{\sinh^2(x)}$	$\int f(x)\,dx = x - \log(\tanh(x)+1) + \log(\tanh(x))$
$f(x) = \operatorname{erf}(x)$	$f'(x) = \frac{2}{\sqrt{\pi}}e^{-x^2}$	$\int f(x)\,dx = x\operatorname{erf}(x) + \frac{e^{-x^2}}{\sqrt{\pi}}$
$f(x) = \operatorname{erfc}(x)$	$f'(x) = -\frac{2}{\sqrt{\pi}}e^{-x^2}$	$\int f(x)\,dx = x\operatorname{erfc}(x) - \frac{e^{-x^2}}{\sqrt{\pi}}$
$f(x) = e^x$	$f'(x) = e^x$	$\int f(x)\,dx = e^x$
$f(x) = \Gamma(x)$	$f'(x) = \Gamma(x)\operatorname{polygamma}(0,x)$	$\int f(x)\,dx = \int \Gamma(x)\,dx$
$f(x) = \log(x)$	$f'(x) = \frac{1}{x}$	$\int f(x)\,dx = x\log(x) - x$
$f(x) = \sin^2(x)$	$f'(x) = 2\sin(x)\cos(x)$	$\int f(x)\,dx = \frac{x}{2} - \frac{1}{2}\sin(x)\cos(x)$

$f(x) = \sin(x)$	$f'(x) = \cos(x)$	$\int f(x)\,dx = -\cos(x)$
$f(x) = \sinh(x)$	$f'(x) = \cosh(x)$	$\int f(x)\,dx = \cosh(x)$
$f(x) = \sqrt{x}$	$f'(x) = \frac{1}{2\sqrt{x}}$	$\int f(x)\,dx = \frac{2x^{\frac{3}{2}}}{3}$
$f(x) = \tan^2(x)$	$f'(x) = \left(2\tan^2(x) + 2\right)\tan(x)$	$\int f(x)\,dx = -x + \frac{\sin(x)}{\cos(x)}$
$f(x) = \tan(x)$	$f'(x) = \tan^2(x) + 1$	$\int f(x)\,dx = -\frac{1}{2}\log\left(\sin^2(x) - 1\right)$
$f(x) = \tanh(x)$	$f'(x) = -\tanh^2(x) + 1$	$\int f(x)\,dx = x - \log(\tanh(x) + 1)$
$f(x) = \csc(x)$	$f'(x) = -\cot(x)\csc(x)$	$\int f(x)\,dx = \frac{1}{2}\log(\cos(x) - 1) - \frac{1}{2}\log(\cos(x) + 1)$

```latex
% pytex23
% nach Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
%%% pyblock, Modul sympy, Funktionen, Ableitungen, Integrale, derivative, latex, eval
\documentclass[parskip=half,fontsize=10,DIV=9,paper=a4]{scrartcl}
\usepackage{pythontex}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}      % Mathematik à la AMS

\begin{document}
\section*{PythonTeX: pyblock, sympy, Funktionen, Ableitungen, Integrale}

\begin{pyblock}
from sympy import *                             # symbolische Mathematik
var('x')                                        # symbolische Variable x

# Liste von Funktionen, die in der Tabelle aufgeführt werden sollen
funktionen = ['acos(x)', 'acosh(x)', 'acot(x)', 'acoth(x)', 'asin(x)', 'asinh(x)',
             'atan(x)', 'atanh(x)', 'cos(x)**2', 'cos(x)', 'cosh(x)', 'cot(x)**2',
             'cot(x)', 'coth(x)', 'erf(x)', 'erfc(x)', 'exp(x)', 'gamma(x)','ln(x)',
             'sin(x)**2', 'sin(x)', 'sinh(x)', 'sqrt(x)', 'tan(x)**2', 'tan(x)',
             'tanh(x)', 'csc(x)']

f          = "f(x) "
fableit    = "f'(x)"
fint       = r"\int\!\!f(x)\,dx"

print(r'\begin{align*}')                          # Tabelle
for funk in funktionen:                           # Schleife
    if funk in ['sin(x)**2']:                     # ggf. neue Seite
        print(r"\displaybreak")
    meineableit = 'Derivative(' + funk + ', x)'   # Ableitung
    meinint     = 'Integral(' + funk + ', x)'     # Integral
    print(latex(f),       r"&=", latex(eval(funk)),                    r'&')
    print(latex(fableit), r"&=", latex(eval(meineableit + '.doit()')), r"&")
    print(latex(fint),    r"&=", latex(eval(meinint + '.doit()')),     r'\\')
print(r'\end{align*}')
\end{pyblock}
\printpythontex

\end{document}
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
# python23.py
# nach Mertz, Slough 2013 - A Gentle Introduction to PythonTeX
# Modul sympy, Funktionen, Ableitungen, Integrale, derivative, latex, eval
print(r"\documentclass[parskip=half,fontsize=10,DIV=9,paper=a4]{scrartcl}")
print(r"\usepackage{fontspec}")
print(r"\usepackage{amsmath}      % Mathematik à la AMS")

print(r"\begin{document}")
print(r"\section*{Python23: sympy, Funktionen, Ableitungen, Integrale}")

from sympy import *                              # symbolische Mathematik
var('x')                                         # symbolische Variable x

# Liste von Funktionen, die in der Tabelle aufgeführt werden sollen
funktionen = ['acos(x)', 'acosh(x)', 'acot(x)', 'acoth(x)', 'asin(x)', 'asinh(x)',
              'atan(x)', 'atanh(x)', 'cos(x)**2', 'cos(x)', 'cosh(x)', 'cot(x)**2',
              'cot(x)', 'coth(x)', 'erf(x)', 'erfc(x)', 'exp(x)', 'gamma(x)','ln(x)',
              'sin(x)**2', 'sin(x)', 'sinh(x)', 'sqrt(x)', 'tan(x)**2', 'tan(x)',
              'tanh(x)', 'csc(x)']

f           = "f(x) "
fableit     = "f'(x)"
fint        = r"\int\!\!f(x)\,dx"

print(r'\begin{align*}')                         # Tabelle
for funk in funktionen:                          # Schleife
    if funk in ['sin(x)**2']:                    # ggf. neue Seite
        print(r"\displaybreak")
    meineableit = 'Derivative(' + funk + ', x)'  # Ableitung
    meinint     = 'Integral(' + funk + ', x)'    # Integral
    print(latex(f),       r"&=", latex(eval(funk)),                    r'&')
    print(latex(fableit), r"&=", latex(eval(meineableit + '.doit()')), r"&")
    print(latex(fint),    r"&=", latex(eval(meinint + '.doit()')),     r'\\')
print(r'\end{align*}')

print(r"\end{document}")
```

```latex
\documentclass[parskip=half,fontsize=10,DIV=9,paper=a4]{scrartcl}
\usepackage{fontspec}
\usepackage{amsmath}      % Mathematik à la AMS
\begin{document}
\section*{Python23: sympy, Funktionen, Ableitungen, Integrale}
\begin{align*}
f(x)  &= \operatorname{acos}{\left (x \right )} &
f'(x) &= - \frac{1}{\sqrt{- x^{2} + 1}} &
\int\!\!f(x)\,dx &= x \operatorname{acos}{\left (x \right )} - \sqrt{- x^{2} + 1} \\
f(x)  &= \operatorname{acosh}{\left (x \right )} &
f'(x) &= \frac{1}{\sqrt{x^{2} - 1}} &
\int\!\!f(x)\,dx &= x \operatorname{acosh}{\left (x \right )} - \sqrt{x^{2} - 1} \\
f(x)  &= \operatorname{acot}{\left (x \right )} &
f'(x) &= - \frac{1}{x^{2} + 1} &
\int\!\!f(x)\,dx &= x \operatorname{acot}{\left (x \right )} + \frac{1}{2} \log{\left (x^{
f(x)  &= \operatorname{acoth}{\left (x \right )} &
f'(x) &= \frac{1}{- x^{2} + 1} &
\int\!\!f(x)\,dx &= x \operatorname{acoth}{\left (x \right )} + \log{\left (x + 1 \right )
f(x)  &= \operatorname{asin}{\left (x \right )} &
f'(x) &= \frac{1}{\sqrt{- x^{2} + 1}} &
\int\!\!f(x)\,dx &= x \operatorname{asin}{\left (x \right )} + \sqrt{- x^{2} + 1} \\
f(x)  &= \operatorname{asinh}{\left (x \right )} &
f'(x) &= \frac{1}{\sqrt{x^{2} + 1}} &
\int\!\!f(x)\,dx &= x \operatorname{asinh}{\left (x \right )} - \sqrt{x^{2} + 1} \\
f(x)  &= \operatorname{atan}{\left (x \right )} &
f'(x) &= \frac{1}{x^{2} + 1} &
\int\!\!f(x)\,dx &= x \operatorname{atan}{\left (x \right )} - \frac{1}{2} \log{\left (x^{
f(x)  &= \operatorname{atanh}{\left (x \right )} &
f'(x) &= \frac{1}{- x^{2} + 1} &
\int\!\!f(x)\,dx &= x \operatorname{atanh}{\left (x \right )} + \log{\left (x + 1 \right )
f(x)  &= \cos^{2}{\left (x \right )} &
f'(x) &= - 2 \sin{\left (x \right )} \cos{\left (x \right )} &
\int\!\!f(x)\,dx &= \frac{x}{2} + \frac{1}{2} \sin{\left (x \right )} \cos{\left (x \right
f(x)  &= \cos{\left (x \right )} &
f'(x) &= - \sin{\left (x \right )} &
\int\!\!f(x)\,dx &= \sin{\left (x \right )} \\
f(x)  &= \cosh{\left (x \right )} &
f'(x) &= \sinh{\left (x \right )} &
\int\!\!f(x)\,dx &= \sinh{\left (x \right )} \\
f(x)  &= \cot^{2}{\left (x \right )} &
f'(x) &= \left(- 2 \cot^{2}{\left (x \right )} - 2\right) \cot{\left (x \right )} &
\int\!\!f(x)\,dx &= - x - \frac{\cos{\left (x \right )}}{\sin{\left (x \right )}} \\
f(x)  &= \cot{\left (x \right )} &
f'(x) &= - \cot^{2}{\left (x \right )} - 1 &
```

```
\int\!\!f(x)\,dx &= \frac{1}{2} \log{\left (\cos^{2}{\left (x \right )} - 1 \right )} \\
f(x)  &= \coth{\left (x \right )} &
f'(x) &= - \frac{1}{\sinh^{2}{\left (x \right )}} &
\int\!\!f(x)\,dx &= x - \log{\left (\tanh{\left (x \right )} + 1 \right )} + \log{\left (\
f(x)  &= \operatorname{erf}{\left (x \right )} &
f'(x) &= \frac{2}{\sqrt{\pi}} e^{- x^{2}} &
\int\!\!f(x)\,dx &= x \operatorname{erf}{\left (x \right )} + \frac{e^{- x^{2}}}{\sqrt{\pi
f(x)  &= \operatorname{erfc}{\left (x \right )} &
f'(x) &= - \frac{2}{\sqrt{\pi}} e^{- x^{2}} &
\int\!\!f(x)\,dx &= x \operatorname{erfc}{\left (x \right )} - \frac{e^{- x^{2}}}{\sqrt{\p
f(x)  &= e^{x} &
f'(x) &= e^{x} &
\int\!\!f(x)\,dx &= e^{x} \\
f(x)  &= \Gamma{\left(x \right)} &
f'(x) &= \Gamma{\left(x \right)} \operatorname{polygamma}{\left (0,x \right )} &
\int\!\!f(x)\,dx &= \int \Gamma{\left(x \right)}\, dx \\
f(x)  &= \log{\left (x \right )} &
f'(x) &= \frac{1}{x} &
\int\!\!f(x)\,dx &= x \log{\left (x \right )} - x \\
\displaybreak
f(x)  &= \sin^{2}{\left (x \right )} &
f'(x) &= 2 \sin{\left (x \right )} \cos{\left (x \right )} &
\int\!\!f(x)\,dx &= \frac{x}{2} - \frac{1}{2} \sin{\left (x \right )} \cos{\left (x \right
f(x)  &= \sin{\left (x \right )} &
f'(x) &= \cos{\left (x \right )} &
\int\!\!f(x)\,dx &= - \cos{\left (x \right )} \\
f(x)  &= \sinh{\left (x \right )} &
f'(x) &= \cosh{\left (x \right )} &
\int\!\!f(x)\,dx &= \cosh{\left (x \right )} \\
f(x)  &= \sqrt{x} &
f'(x) &= \frac{1}{2 \sqrt{x}} &
\int\!\!f(x)\,dx &= \frac{2 x^{\frac{3}{2}}}{3} \\
f(x)  &= \tan^{2}{\left (x \right )} &
f'(x) &= \left(2 \tan^{2}{\left (x \right )} + 2\right) \tan{\left (x \right )} &
\int\!\!f(x)\,dx &= - x + \frac{\sin{\left (x \right )}}{\cos{\left (x \right )}} \\
f(x)  &= \tan{\left (x \right )} &
f'(x) &= \tan^{2}{\left (x \right )} + 1 &
\int\!\!f(x)\,dx &= - \frac{1}{2} \log{\left (\sin^{2}{\left (x \right )} - 1 \right )} \\
f(x)  &= \tanh{\left (x \right )} &
f'(x) &= - \tanh^{2}{\left (x \right )} + 1 &
\int\!\!f(x)\,dx &= x - \log{\left (\tanh{\left (x \right )} + 1 \right )} \\
f(x)  &= \csc{\left (x \right )} &
f'(x) &= - \cot{\left (x \right )} \csc{\left (x \right )} &
\int\!\!f(x)\,dx &= \frac{1}{2} \log{\left (\cos{\left (x \right )} - 1 \right )} - \frac{
```

```
\end{align*}
\end{document}
```