

Lösungsüberprüfung mit Python/sympy

Um Aufgaben zu lösen, bei denen es um z.B. Ausklammern oder Vereinfachen geht, kann man Python mit dem Module `sympy` verwenden.

Bei der Eingabe der Terme muss man auf die korrekte Syntax achten. Anders als in der Mathematik üblich, muss zwischen jedem Operanden ein Operator stehen:

x y muss als $x*y$ geschrieben werden

x^2 muss als $x**2$ geschrieben werden

[simplify.py](#)

```
#!/usr/bin/env python3

from sympy import *

# hier werden die Symbole, als die Variablen benannt. Fehlt eine Variable, so kann man diese hinzufügen oder eine bereits bekannte nutzen
x, y, z, a, b, c, u, v, m, n = symbols('x y z a b c u v m n')

# Um die Ausgabe etwas schöner zu machen
init_printing(use_unicode=True)

# Man übergibt die Funktion als funktion und den Namen, den sie in der Ausgabe erhalten soll
# als Ergebnis erhält man eine Ausdruck, der sich direkt per Copy&Paste in Libreoffice als Formel einfügen lässt
def get_odt_of(funktion, name):
    funktion= mathematica_code(simplify(funktion))
    funktion= funktion.replace("*", " cdot ")
    funktion= funktion.replace(".", ",")
    funktion= funktion.replace(",0", "")
    funktion= funktion.replace("Log", "log")
    funktion= funktion.replace("[", "(")
    funktion= funktion.replace("]", ")")
    funktion= funktion.replace("{", "")
    funktion= funktion.replace("}", "")
    funktion= funktion.replace("/", "over")
    print ( name+(x)=", funktion)
    return funktion

# Man übergibt die Funktion als funktion und den Namen, den sie in der Ausgabe erhalten soll
# als Ergebnis erhält man eine Ausdruck, der sich direkt per Copy&Paste in Geobra als Formel einfügen lässt
```

```

def get_geogebra_of(funktion, name):
    funktion= mathematica_code(simplify(funktion))
    funktion= funktion.replace(".0", "")
    funktion= funktion.replace("Log", "log")
    funktion= funktion.replace("[", "(")
    funktion= funktion.replace("]", ")")
    funktion= funktion.replace("{", "")
    funktion= funktion.replace("}", "")
    print ( name+(x)=", funktion)
    return funktion

# Beispiel: Der Ausdruck q soll ausgeklammert werden
q=(3*a - 5*b) *(6*x - 7*y + 9*z) - (5*x-8*y +8*z)*(4*a-5*b)
get_odt_of(sympify(q), "q")

```

Ausgabe in Idle:

```
q(x)= -2 cdot a cdot x + 11 cdot a cdot y - 5 cdot a cdot z - 5 cdot b cdot x - 5 cdot b cdot y - 5 cdot b cdot z
```

Nach Copy&Paste in LibreOffice:

$$q(x) = -2 \cdot a \cdot x + 11 \cdot a \cdot y - 5 \cdot a \cdot z - 5 \cdot b \cdot x - 5 \cdot b \cdot y - 5 \cdot b \cdot z$$

Mit Live SymPy direkt ausprobieren

Live SymPy ist eine Seite, auf der man seine Eingabe direkt vornehmen kann. D.h. man muss nichts installieren und kann für Kleinigkeiten direkt loslegen.

Hinweis: Unter Umständen muss man seine eigenen Variablen hinzufügen oder auf die bereits existierenden umbenennen. Die folgende Code-Zeilen sind für die Variablen bzw. Funktionsnamen zuständig. In SymPy werden diese `symbols` genannt:

```
x, y, z, t = symbols('x y z t')
k, m, n = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)
```

Dabei sind `x`, `y`, `z` und `t` Variablen. `k`, `m` und `n` sind ganzzahlige (`integer=True`) Laufvariablen für z.B. Summenausdrücke. Und zum Schluss die Funktionen `f`, `g` und `h` als Klasse `cls=Function`.

From:

<http://www.kopfload.de/> - kopfload - Lad Dein Hirn auf!

Permanent link:

<http://www.kopfload.de/doku.php?id=lager:mathe:python&rev=1481129568>

Last update: **2025/11/19 16:13**



