

WolframAlpha



Wolfram Alpha é uma linguagem que oferece um mecanismo de conhecimento computacional, desenvolvido pela empresa Wolfram Research.



Mathematics ›

 Step-by-Step Solutions

 Elementary Math

x^2-1 Algebra

 Plotting & Graphics

$\int f(x) dx$ Calculus & Analysis

 Geometry

$y''(x)$ Differential Equations

 Statistics

 More Topics ›

Science & Technology ›

 Units & Measures

 Physics

 Chemistry

 Engineering

 Computational Sciences

 Earth Sciences

 Materials

 Transportation

 More Topics ›

Society & Culture ›

 People

 Arts & Media

 Dates & Times

 Words & Linguistics

 Money & Finance

 Food & Nutrition

 Political Geography

 History

 More Topics ›

Everyday Life ›

 Personal Health

 Personal Finance


 Surprises

 Entertainment

 Household Science

 Household Math

 Hobbies

 Today's World

 More Topics ›



Wolfram Mathematica

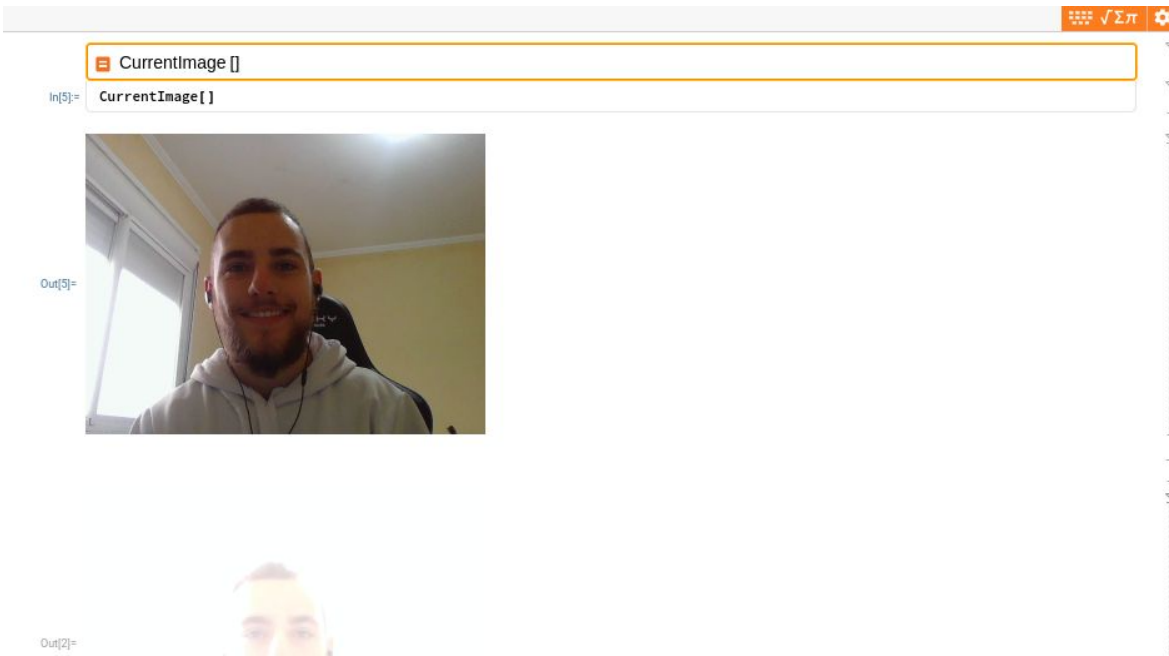
Grande parte da sua popularidade vem de sua parte matemática, conhecida como Wolfram Mathematica, uma plataforma computacional que oferece ilimitados recursos ligados a diversas áreas, como álgebra, matemática discreta, gráficos, estatística, entre diversas outras. Esses recursos são fortemente baseados em programação funcional.



Brincando com Wolfram Alpha

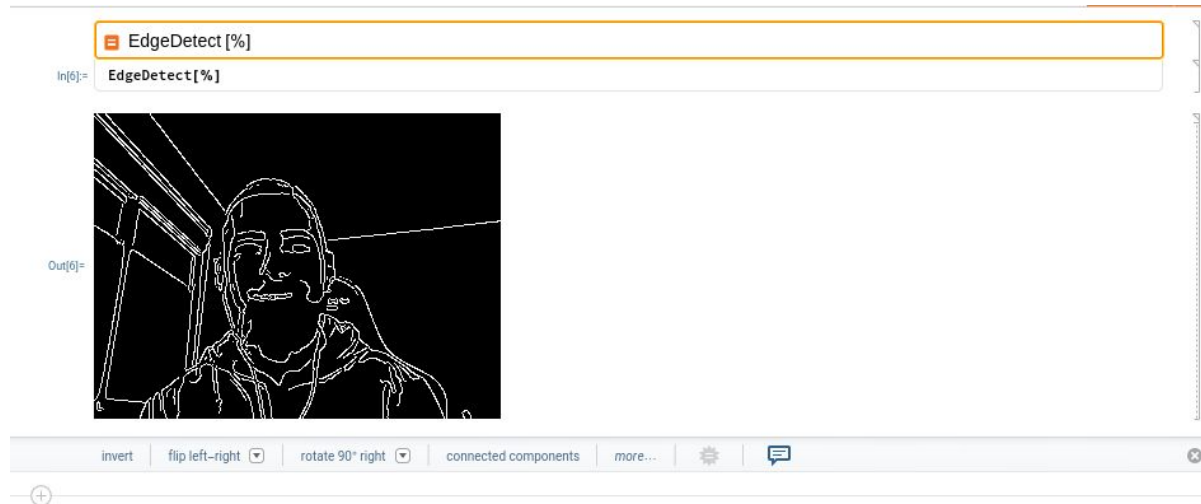
Em uma primeira visão da plataforma, estaremos usando o Wolfram Alpha Notebook para fazer experimentos interessantes, que usam códigos já feitos na base de dados do serviço.

Brincando com Wolfram Alpha



The screenshot displays the Wolfram Alpha web interface. At the top, there is a navigation bar with icons for a grid, mathematical symbols ($\sqrt{\Sigma\pi}$), and a settings gear. Below this, the input field contains the text `CurrentImage []`. The output area, labeled `Out[5]=`, shows a video feed of a man with a beard and headphones, wearing a white hoodie, sitting in front of a window. The video feed is displayed within a rectangular frame. Below the video feed, there is a smaller, partially visible output labeled `Out[2]=` showing a close-up of a person's face.

Brincando com Wolfram Alpha

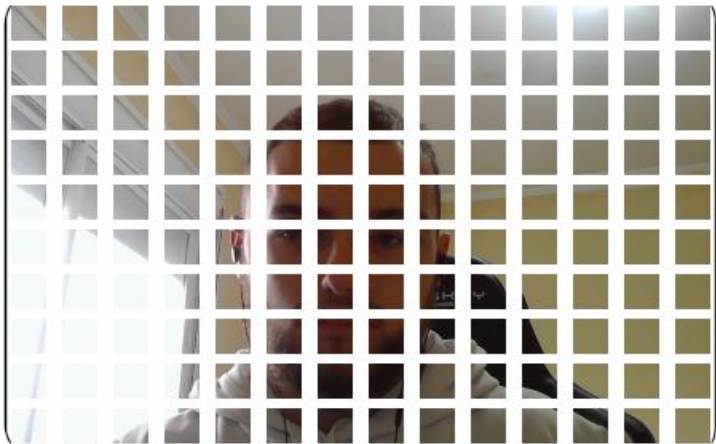


Brincando com Wolfram Alpha

```
ImagePartition[CurrentImage[], 22]
```

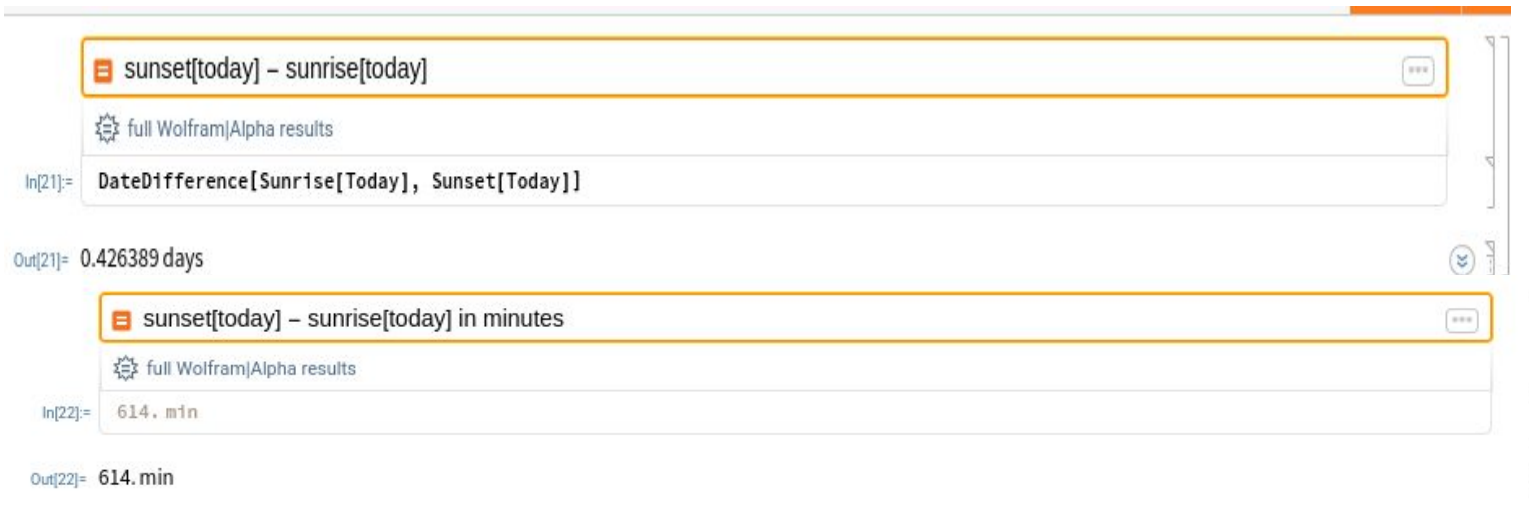
```
In[9]:= ImagePartition[CurrentImage[], 22]
```

```
Out[9]=
```



Brincando com Wolfram Alpha

Wolfram usa NLP (Natural Language Processing, ou Processamento de Linguagem Natural, campo da inteligência artificial), para interpretar comandos do terminal, veja:



The screenshot displays the Wolfram Alpha interface with two input queries and their corresponding outputs. The first query, "sunset[today] - sunrise[today]", is entered into the top input box. Below it, the output is shown as "Out[21]= 0.426389 days". The second query, "sunset[today] - sunrise[today] in minutes", is entered into the bottom input box. Below it, the output is shown as "Out[22]= 614. min". The interface includes a search bar, a settings icon, and a "full Wolfram|Alpha results" link for each query.

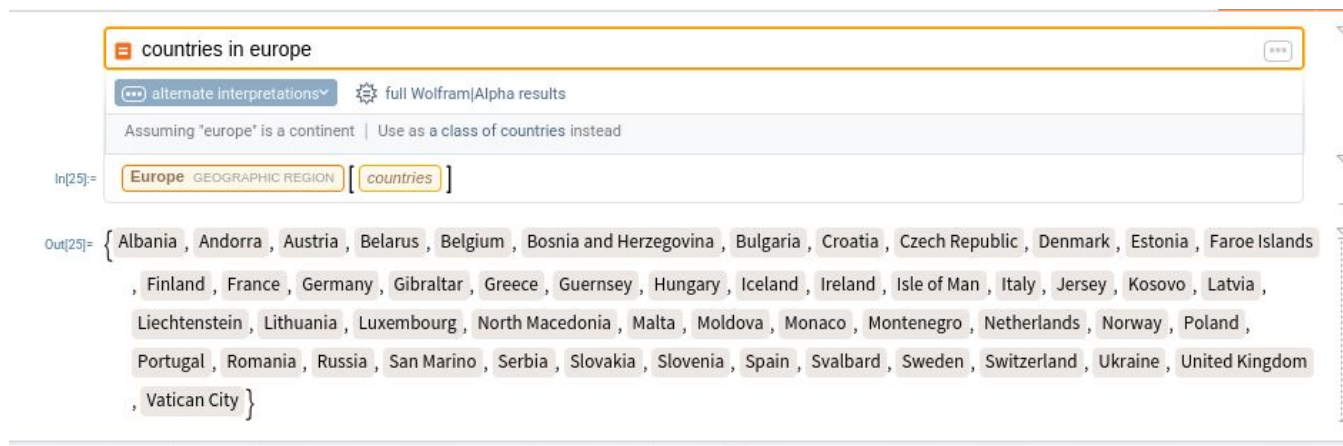
Input 1: `sunset[today] - sunrise[today]`

Output 1: `Out[21]= 0.426389 days`

Input 2: `sunset[today] - sunrise[today] in minutes`

Output 2: `Out[22]= 614. min`

Brincando com Wolfram Alpha



The screenshot shows the Wolfram Alpha interface. At the top, the input bar contains the text "countries in europe". Below the input bar, there are two buttons: "alternate interpretations" and "full Wolfram|Alpha results". Below these buttons, there is a line of text: "Assuming 'europe' is a continent | Use as a class of countries instead". Below this line, the input is shown as `In[25]= Europe GEOGRAPHIC REGION [countries]`. The output is shown as `Out[25]= { Albania , Andorra , Austria , Belarus , Belgium , Bosnia and Herzegovina , Bulgaria , Croatia , Czech Republic , Denmark , Estonia , Faroe Islands , Finland , France , Germany , Gibraltar , Greece , Guernsey , Hungary , Iceland , Ireland , Isle of Man , Italy , Jersey , Kosovo , Latvia , Liechtenstein , Lithuania , Luxembourg , North Macedonia , Malta , Moldova , Monaco , Montenegro , Netherlands , Norway , Poland , Portugal , Romania , Russia , San Marino , Serbia , Slovakia , Slovenia , Spain , Svalbard , Sweden , Switzerland , Ukraine , United Kingdom , Vatican City }`.

countries in europe

alternate interpretations full Wolfram|Alpha results

Assuming "europe" is a continent | Use as a class of countries instead

In[25]= Europe GEOGRAPHIC REGION [countries]

Out[25]= { Albania , Andorra , Austria , Belarus , Belgium , Bosnia and Herzegovina , Bulgaria , Croatia , Czech Republic , Denmark , Estonia , Faroe Islands , Finland , France , Germany , Gibraltar , Greece , Guernsey , Hungary , Iceland , Ireland , Isle of Man , Italy , Jersey , Kosovo , Latvia , Liechtenstein , Lithuania , Luxembourg , North Macedonia , Malta , Moldova , Monaco , Montenegro , Netherlands , Norway , Poland , Portugal , Romania , Russia , San Marino , Serbia , Slovakia , Slovenia , Spain , Svalbard , Sweden , Switzerland , Ukraine , United Kingdom , Vatican City }

Brincando com Wolfram Alpha

EntityValue [%,"Flag"]

alternate interpretations full Wolfram|Alpha results

Assuming "%" is referring to a computation Use as a unit instead

In[28]:= EntityValue[%, "Flag"]

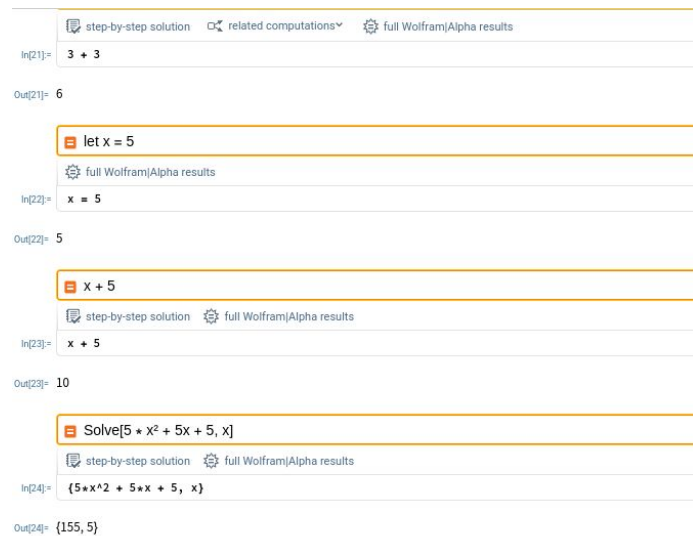
The image shows a screenshot of a Wolfram Alpha interface. At the top, the input field contains the query "EntityValue [%,"Flag"]". Below the input field, there are two tabs: "alternate interpretations" and "full Wolfram|Alpha results". The "full Wolfram|Alpha results" tab is selected. Below the tabs, there is a line of text: "Assuming "%" is referring to a computation Use as a unit instead". Below this, the input field is repeated as "In[28]:= EntityValue[%, "Flag"]". The results are displayed as a collection of 18 national flags arranged in three rows. The flags are: Albania, Spain, Austria, Serbia, Belgium, Bosnia and Herzegovina, Bulgaria, Croatia, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, and Italy.



Wolfram Mathematica e a Programação Funcional

Como já dito antes, a parte matemática da linguagem Wolfram apresenta grande suporte à programação funcional. Veremos agora alguns usos diretos.

Experimentos com Wolfram Mathematica



The screenshot displays the Wolfram Mathematica interface with four input-output sessions. Each session includes a toolbar with icons for 'step-by-step solution', 'related computations', and 'full Wolfram|Alpha results'.

Session 1:
In[21]:= $3 + 3$
Out[21]= 6

Session 2:
In[22]:= `let X = 5`
Out[22]= 5

Session 3:
In[23]:= $x + 5$
Out[23]= 10

Session 4:
In[24]:= `Solve[5 + x^2 + 5x + 5, x]`
Out[24]= $\{155, 5\}$

Aqui observamos usos mais básicos, como somar, atribuir valores a variáveis, e resolver equações

Experimentos com Wolfram Mathematica



The screenshot displays the Wolfram Mathematica interface with three input-output sessions. Each session includes a text input field, a settings bar with icons for 'full Wolfram|Alpha results', 'step-by-step solution', and 'related computations', and a code input field.

Session 1:

- Input: `let f(y) = y + 5y - sin(y)`
- Code: `f[y_] := y + 5*y - Sin[y]`
- Output: `192 - sin(32)`

Session 2:

- Input: `f(32)`
- Code: `f[32]`
- Output: `3 y^2 + cos(y) + c1`

Session 3:

- Input: `integrate f(y)`
- Code: `Integrate[f[y], y]`
- Output: `3 y^2 + cos(y) + c1`

Session 4:

- Input: `now integrate it again`
- Code: `Integrate[3*y^2 + C[1] + Cos[y], y]`
- Output: `y^3 + sin(y) + c1 y + c2`

Aqui já podemos ver usos mais complexos da programação funcional, como declaração de funções, uso da função default “integrate”, assim como a inteligência artificial de reconhecimento de linguagem a pleno vapor!

Experimentos com Wolfram Mathematica

Out[28]= $3y^2 + \cos(y) + c_1$

now integrate it again

related computations full Wolfram|Alpha results

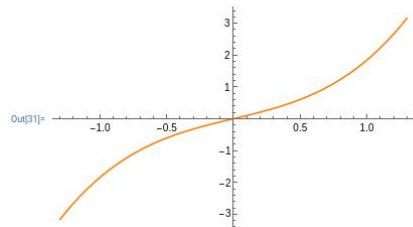
In[29]= `Integrate[3*y^2 + C[1] + Cos[y], y]`

Out[29]= $y^3 + \sin(y) + c_1 y + c_2$

plot Out 29

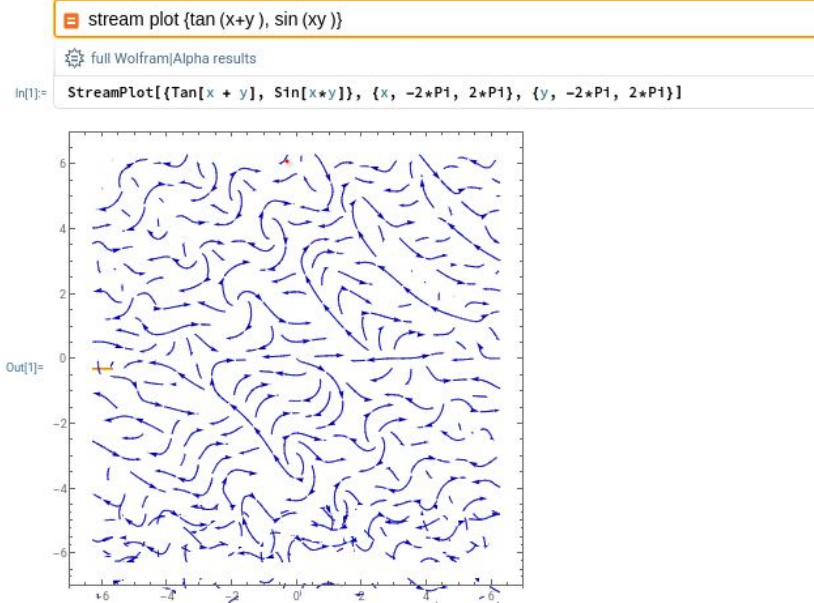
related computations full Wolfram|Alpha results

In[31]= `Plot[Evaluate[y^3 + y*C[1] + C[2] + Sin[y] /. _C -> 0], {y, -1.3, 1.3}]`



Aqui, fornecemos uma uma instrução em linguagem natural, que diz: “Faça o gráfico do resultado 29”. A linguagem transforma isso em uma instrução, onde a função “Plot” recebe como argumento uma lista e uma outra função, algo que é sempre muito presente em qualquer linguagem com suporte à programação funcional.

Experimentos com Wolfram Mathematica



Aqui, utilizamos o paradigma funcional para fazer o gráfico de uma função extremamente complexa.

Experimentos com Wolfram Mathematica

Out[3]= {1, 3, 4, 5, 10}

```
let f(x) = x + 2
```

full Wolfram|Alpha results

In[4]:= f[x_] := x + 2

```
f(a)
```

full Wolfram|Alpha results

In[5]:= f[a]

Out[5]= {3, 5, 6, 7, 12}

Uso similar ao que em Haskell é a função de alta ordem “map”.

Aqui, aplicamos a função $f(x)$ à todos os valores de uma lista.

Experimentos com Wolfram Mathematica

Aninhamento de funções
puramente funcional.

let $f(x) = x^3$

full Wolfram|Alpha results

In[3] = $f[x_] := x^3$

let $g(x) = x * 4$

full Wolfram|Alpha results

In[4] = $g[x_] := x * 4$

let $h(x) = x / 3$

full Wolfram|Alpha results

In[5] = $h[x_] := x/3$

$f(g(h(21)))$

alternate interpretations full Wolfram|Alpha results

Assuming "(" is referring to math | Use "(21)" as referring to math instead

In[6] = $f[g[h[21]]]$

Out[6] = 21952

Experimentos com Wolfram Mathematica

```
let b = {1, 2, 3, 4, 5, 6}
```

full Wolfram|Alpha results

```
In[7]:= b = {1, 2, 3, 4, 5, 6}
```

```
Out[7]= {1, 2, 3, 4, 5, 6}
```

```
f(g(h(b)))
```

alternate interpretations

full Wolfram|Alpha results

Assuming "g" is referring to math | Use as a unit instead

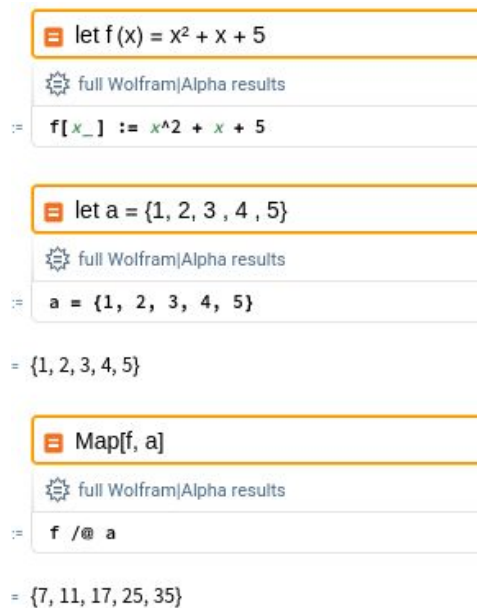
Assuming "h" is referring to math | Use as a unit instead

```
In[8]:= f[g[h[b]]]
```

```
Out[8]= {64/27, 512/27, 64, 4096/27, 8000/27, 512}
```

Mais um uso similar ao “map”,
agora com funções aninhadas.

Experimentos com Wolfram Mathematica



The image shows three code cells from the Wolfram Mathematica interface, each with an input field, a 'full Wolfram|Alpha results' button, and an output field.

Cell 1:
Input: `let f(x) = x^2 + x + 5`
Output: `f[x_] := x^2 + x + 5`

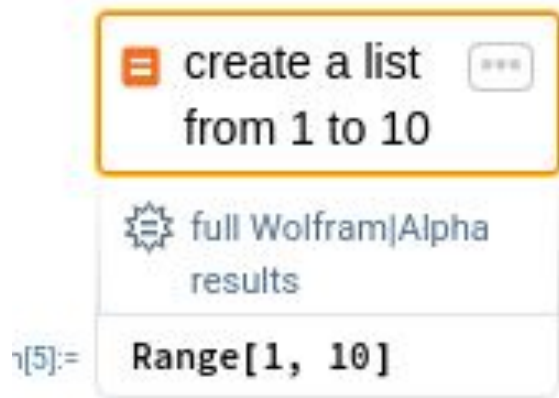
Cell 2:
Input: `let a = {1, 2, 3, 4, 5}`
Output: `a = {1, 2, 3, 4, 5}`

Cell 3:
Input: `Map[f, a]`
Output: `f /@ a`

Below the third cell, the final result is shown: `{7, 11, 17, 25, 35}`

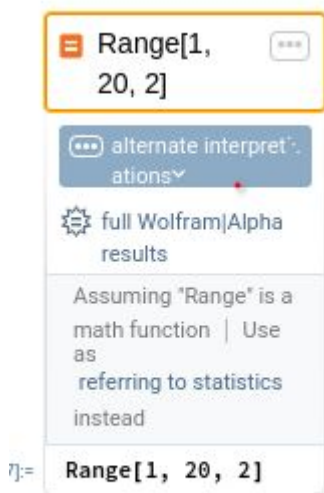
Usando map com função e lista.

Experimentos com Wolfram Mathematica



List comprehension em Wolfram
(Linguagem natural) e sintaxe
padrão

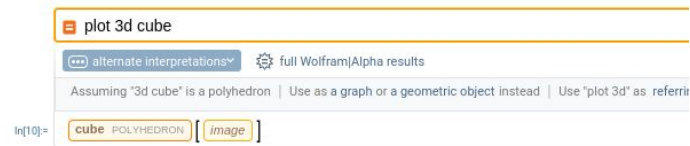
Experimentos com Wolfram Mathematica



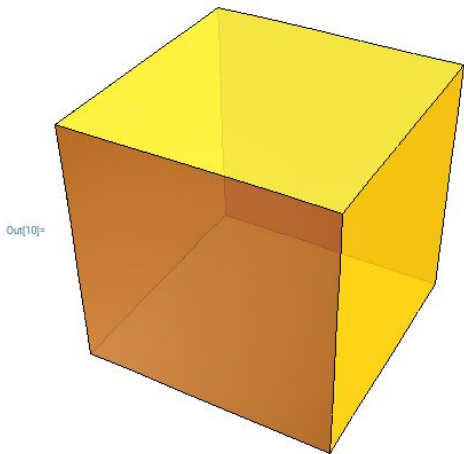
`Range[1, 20, 2]`
`{1, 3, 5, 7, 9, 11, 13, 15, 17, 19}`

List comprehension em Wolfram
em sintaxe padrão

Experimentos com Wolfram Mathematica



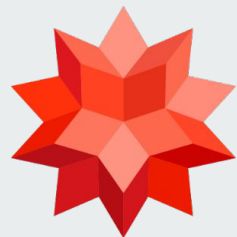
Plotando gráfico 3D com linguagem natural.





Desvantagens do Wolfram

- É licenciado e pago.
- O custo computacional é gigantesco, criando muitas limitações.



WolframAlpha