WolfBot

An introductory guide

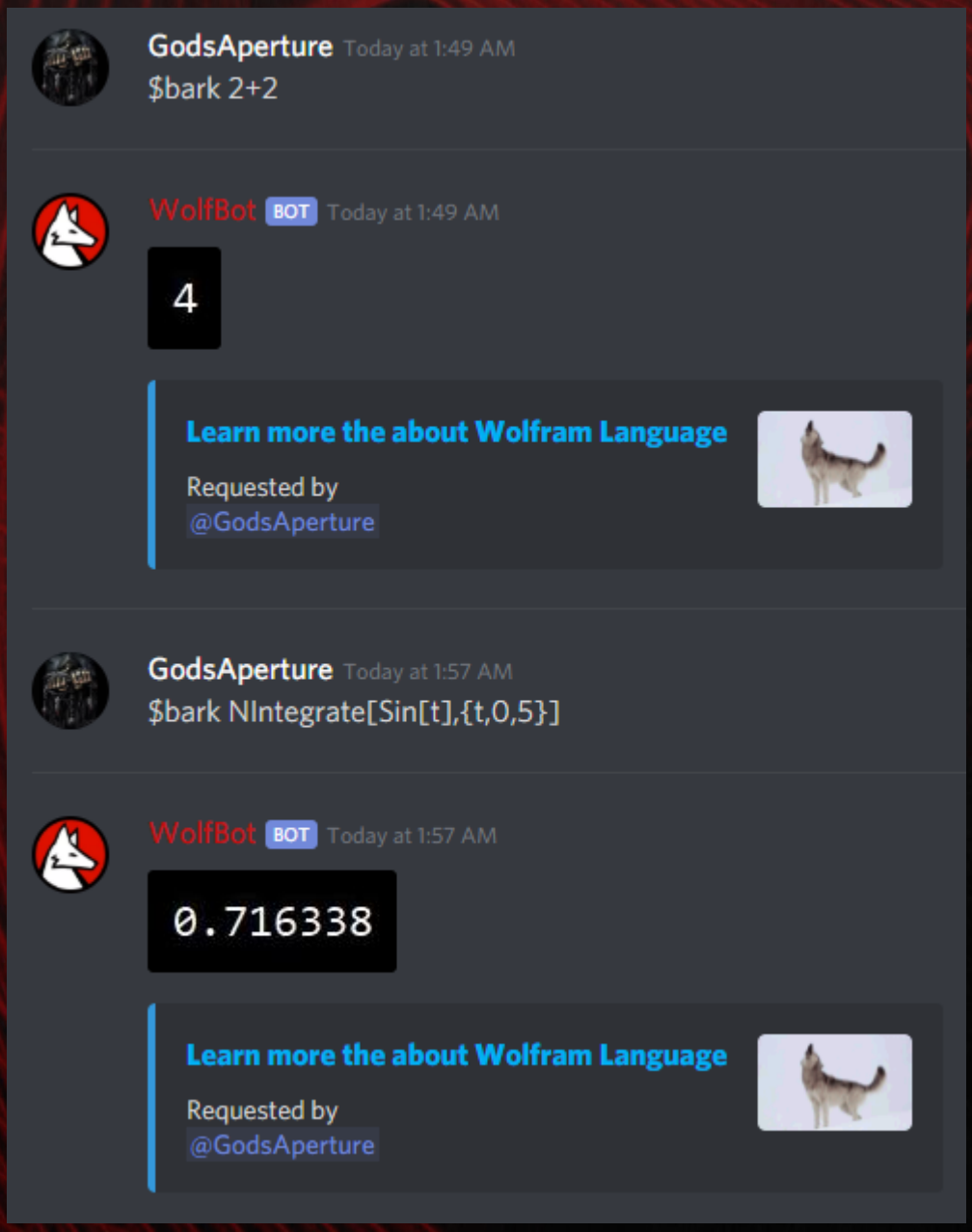
Version 1.0

References

Welcome to WolfBot! Here, you will learn the fundamentals of the Wolfram programming language. This document will cover basics such as declaring, delayed declaring, functions, calculus, and some replacement rules.

Commands

There are a few commands for the WolfBot, $bark will send your input to WolfBot for evaluation in the Wolfram programming language. $bark, $help, $alpha, and $docs are the four currently existing functions.



Declarations

Declarations come in two forms in the Wolfram language. There's the standard declaration, known as the *Set*, and then there is the delayed version, known as *SetDelayed*. The difference is that the right-hand side of the *SetDelayed* function y2 is unevaluated until it is called upon.

In this given example, y2 is the only variable that has a different output because of *SetDelayed*.

Set takes the form of = or Set[x,6] in the first line. *SetDelayed* takes the form := or SetDelayed[y2, x^2] in the third line of code.

```
In[1]:= x = 6;

In[2]:= y1 = x^2;
        y2 := x^2;

In[4]:= y1

Out[4]= 36

In[5]:= y2

Out[5]= 36

In[6]:= x = 4;

In[7]:= y1

Out[7]= 36

In[8]:= y2

Out[8]= 16
```

Boolean Algebra

True and false statements are bread and butter to logic in programming. True and false statements use the == or *Equal* function to check for trueness.

```
In[1]:= 2 + 2 == 4
Out[1]= True

In[2]:= 2 + 3 == 6
Out[2]= False

In[3]:= x == 6
Out[3]= x == 6
```

The third line remains unevaluated because x is undefined, and in some functions like *If* will return a third option since it is neither true nor false.

Functions

Functions are very easy to write in WolfBot. It's very similar to writing functions in mathematics so if you know what a mathematical function is like, then this should be easy to understand and follow.

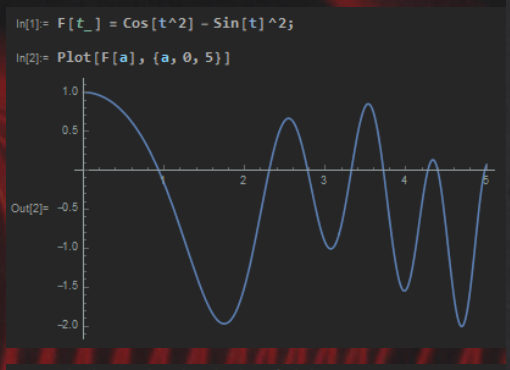
```
In[1]:= F[x_] = Sin[x] + Cos[x]^2 - E^x
Out[1]= -e^x + Cos[x]^2 + Sin[x]

In[2]:= F[2]
Out[2]= -e^2 + Cos[2]^2 + Sin[2]

In[3]:= G[x_, t_] = ArcTan[x, t] + t^2 - x^3
Out[3]= t^2 - x^3 + ArcTan[x, t]

In[4]:= G[1, 2]
Out[4]= 3 + ArcTan[2]
```

Functions can be defined using *Set* or *SetDelayed*. Some functions may require that you use one or the other so it is suggested to learn that *Set* defines the right-hand side immediately after evaluation while *SetDelayed* will only evaluate the right-hand side after the function is being called.

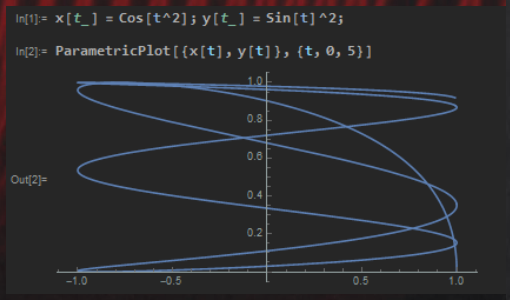
Plotting

Plotting is by far one of the most important abilities of a mathematician, that's why WolfBot will rely on the Wolfram language's plethora of plotting functions.

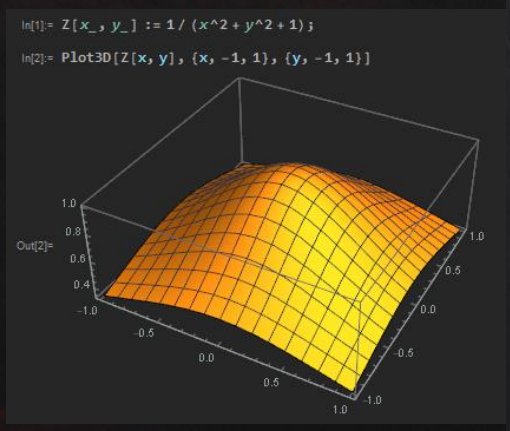
Plot



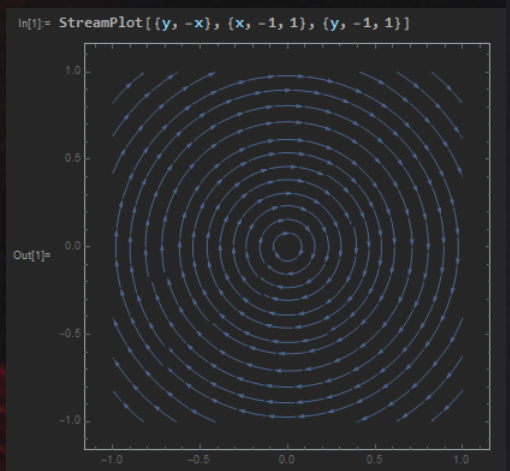
ParametricPlot



Plot3D



StreamPlot



Solving

Solving is a fundamental ability of mathematicians, engineers, scientists, and others. However, at times it can be tedious to solve the same thing or something gargantuan. That's why there's *Solve* and *NSolve*.

Solve will use known methods to manipulate and solve for exact solutions of given equations and conditions.

```
In[1]:= Solve[y / m == x + b / m, y]
Out[1]= {{y → b + m x}}

In[2]:= Solve[y == a x^2 + b x + c, x]
Out[2]= {{x → (-b + Sqrt[b^2 - 4 a c + 4 a y]) / (2 a)}, {x → -(b + Sqrt[b^2 - 4 a c + 4 a y]) / (2 a)}}
```

NSolve will use numerical methods to approximate real, imaginary, and complex solutions to given equations.

```
In[1]:= NSolve[x^2 + 3 x + 2 == 0, x]
Out[1]= {{x → -2.}, {x → -1.}}

In[2]:= NSolve[E^t == t, t]
NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.
Out[2]= {{t → 0.3181315052 - 1.337235701 i}}
```

Matrices and Lists

Matrices and lists in the Wolfram language are the same thing, that means there's no need to specify that a list is a matrix.

Lists can be made either by the *List* function or using curly braces.

```
In[1]:= List[1, 2, 3, 4]
Out[1]= {1, 2, 3, 4}

In[2]:= {1, 2, 3, 4} + a
Out[2]= {1 + a, 2 + a, 3 + a, 4 + a}

In[3]:= Sin[{1, 2, 3, 4}]
Out[3]= {Sin[1], Sin[2], Sin[3], Sin[4]}
```

Matrices are simply lists within lists, and to multiply matrices, you need only put a period between the two matrices. Putting a space will multiply each corresponding entry.

```
In[1]:= A = {{1, 2}, {3, 4}};
        B = {{5, 6}, {7, 8}};
        MatrixForm[A]
Out[3]//MatrixForm=
        ⎛1  2⎞
        ⎝3  4⎠

In[4]:= A B
Out[4]= {{5, 12}, {21, 32}}

In[5]:= A.B
Out[5]= {{19, 22}, {43, 50}}

In[6]:= {{a, b}, {c, d}} {{e, f}, {g, h}}
Out[6]= {{a e, b f}, {c g, d h}}

In[7]:= {{a, b}, {c, d}} .{{e, f}, {g, h}}
Out[7]= {{a e + b g, a f + b h}, {c e + d g, c f + d h}}
```

Calculus

Limits are a vital component to understanding how derivatives and integrals work in calculus, without such a concept, grasping what an integral or a derivative becomes more challenging.

The function *Limit* is used for all limits in the Wolfram language, and even has L'Hopital's Rule programmed to automatically differentiate in the event of an indeterminant value.

```
In[1]:= F[x_] := Piecewise[{{2, x == 1}, {x, x ≠ 1}}];

In[2]:= Limit[F[x], x → 1]

Out[2]= 1

In[3]:= Limit[(Sin[x + h] - Sin[x]) / h, h → 0]

Out[3]= Cos[x]
```

Limit does not attempt to evaluate at the given limit first, it will approach from all directions in the complex plane instead.

```
In[1]:= Limit[Sin[(x - y)^2 / (x^2 + y^2)], {x → 0, y → 0}]

Out[1]= Sin[1]
```

Limit will also perform multivariable limits.

Derivatives

Calculus removes the limited understanding and limitations of equation deriving from algebra. Calculus also allows us to study the behavior of functions and equations.

Derivatives are simple and straightforward in the Wolfram language and can be obtained two different ways in WolfBot.

```
In[1]:= F[x_] := Cos[x] + x^2 - Lap[x];

In[2]:= F'[x]

Out[2]= 2 x - Sin[x] - Lap'[x]

In[3]:= F''[x]

Out[3]= 2 - Cos[x] - Lap''[x]

In[4]:= D[F[x], x]

Out[4]= 2 x - Sin[x] - Lap'[x]

In[5]:= D[F[x], {x, 2}]

Out[5]= 2 - Cos[x] - Lap''[x]
```

Placing an apostrophe after a function of a single variable, whether defined or undefined, will represent the derivative of the function, again whether the function is defined or undefined.

Integrals

Integrals are perhaps one of the most important concepts to grasp in calculus, and have very important applications in topics such as ordinary and partial differential equations.

The Wolfram language has two different functions for integrating. There's *Integrate* and the numerical counterpart *NIntegrate*.

```
In[1]:= Integrate[Cos[x] + x^2, x]

Out[1]= x^3/3 + Sin[x]

In[2]:= Integrate[Cos[x] + x^2, {x, a, b}]

Out[2]= 1/3 (-a^3 + b^3 - 3 Sin[a] + 3 Sin[b])

In[3]:= Integrate[Cos[x] + x^2, {x2, a, b}, {x, 0, x2}]

Out[3]= Cos[a] + 1/12 (-a^4 + b^4 - 12 Cos[b])
```

Relying on the Risch algorithm, Integrate will return an antiderivative, a bounded integral, or even a multiple integral, each with differing bounds. The bounds are read the same way they would be read on an integral in a textbook, the outermost bound is the farthest left in the list.

```
In[1]:= Integrate[Cos[x] + x^2, {x, bottom, top}] // Expand

Out[1]= -bottom^3/3 + top^3/3 - Sin[bottom] + Sin[top]
```

Integrals

Integrals are perhaps one of the most important concepts to grasp in calculus, and have very important applications in topics such as ordinary and partial differential equations.

NIntegreate is the numerical integration function in the Wolfram programming language.

```
In[1]:= NIntegrate[Cos[x] + x^2, {x, 0, 5}]

Out[1]= 40.70774239

In[2]:= NIntegrate[Cos[x] + x^2, {x2, 0, 5}, {x, 0, x2}]

Out[2]= 52.79967107
```

The Wolfram programming language switches between various methods of numerical integration depending on the given integrand in an attempt to approximate the best answer for the given integrang and interval.

Credits

The Wolfram language, Wolfram Alpha, and Mathematica are property of Wolfram Research, we make this bot for convenient access to the programming language and to encourage others to engage in mathematics and programming in a beautiful composite. If you like the Wolfram language or any other of Wolfram Research's products, please consider buying Mathematica or visiting Wolfram.com for their other products and services.

UncannyEngineer#0163
Project Manager

Dagger#8243
Debugger
Assistant Programmer

GodsAperture#8507
Guide Developer
Assistant Programmer

Thanos#3337
Chief Technical Officer
Debugger