

Mathematica

Hebe Huang and Roshani Shrestha





Who is Stephen Wolfram?

- Creator of Mathematica, Wolfram|Alpha, and the Wolfram Language
- Youngest recipient of the MacArthur Fellowship (1981)
- Aimed to make the world's knowledge computable and accessible to everyone





Let's get started!

- Open up a new notebook.
- Add new sections by pressing the down arrow key.
- By default, Mathematica expects an input cell to have Wolfram Language code. You can press the plus button on the left to insert different types of input and text.



Free-Form Input





General things to know...

- The user can type in commands in plain English, which makes it more simple to understand.
- Start with = *userinput*. Press Shift+Enter to signal that you're done inputting commands. Then, it searches for what you've typed.
- You will be able to see a suggestion bar below the output.



Wolfram Language





Some syntax to start...

- Uses symbolic expressions: in the format `head[arguments]`
- Assignment works the same way as in most languages (ie `x=5`; this is also known as immediate assignment). You can also use `:=` to continuously assign values (ie `t := LocalTime[]` will continuously set `t` to the current time in your time zone)
- Rational numbers are represented by fractions, which is useful because it prevents rounding errors that are found in many languages when using decimals. `I` represents imaginary numbers.



Functions!

- Function parameters are placed in brackets [] unlike Python, which uses parenthesis ().
- The first letter of each word in a function is capitalized.
- `f[x_, y_] := x+y`
 - You can also do `f[5] = 9`, and `f[5]` will always return 9



Lists

- Lists use one-based indexing and go in curly braces {}
- This is to differentiate between standard lists, which use zero-based indexing, and Mathematica lists.



Let's now go over some useful functions...

- `Range[n]`
 - Returns a list of numbers from 1 to n
- `ListPlot[{{x1, y1}, ..., {xn, yn}}]`
 - Returns a scatter plot of points with the given coordinates.
- `Print[expr]`
 - Prints *expr* as an output.



Functions (continued)

- `Graphics[primitives, options]`, `Graphics3D[primitives, options]`
 - Returns a 2D or 3D graphical image based on the primitives and options.
- `Sphere[p, r]`
 - Returns a sphere with radius r centered at point p .



Functions (continued)

- `Plot3D[f, {x, xmin, xmax}, {y, ymin, ymax}]`
 - Returns a 3D graph of a function f with minimum and maximum x and y values.
- `TemplateApply[StringTemplate["hello ``, it is now <* Now *>"], {"user"}]`
 - `` inserts values, <...*> evaluates values



Interactive Features

- `Manipulate[item, {n, min, max, step}]`
 - Adds a slider to the item like desmos and links it to the value of `n`
- `TabView[a, b, c, ...]`
 - Takes in a list of items and allows you to switch between them using tabs
- `Button[TextString, action]`
 - Creates a button with `TextString` as the label and links it to an action when the user clicks on it



Applications





Web Development

- Mathematica can be used to create interactive websites.
- You can create embeddable code through `EmbedCode[APIFunction[], language]`.
- Without the *language* argument, `EmbedCode` returns a string to be inserted into an HTML file. You must use `CloudDeploy` on the `APIFunction` for it to work. There must be another argument inserted, which is `Permissions` -> "Public", for the Mathematica object to be viewed by everyone who visits the page.
 - `EmbedCode[CloudDeploy[Print["Hello World!"]], Permissions -> "Public"]`