

$$(X, O) = e^{-\frac{x^2}{2\sigma^2}}$$

$$x(X, O) = -\frac{x}{\sigma^2} G(X, O) = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

$$xx(X, O) = \frac{x^2 - \sigma^2}{\sigma^4} G(X, O) = \frac{x^2 - \sigma^2}{\sigma^4} e^{-\frac{x^2}{2\sigma^2}}$$

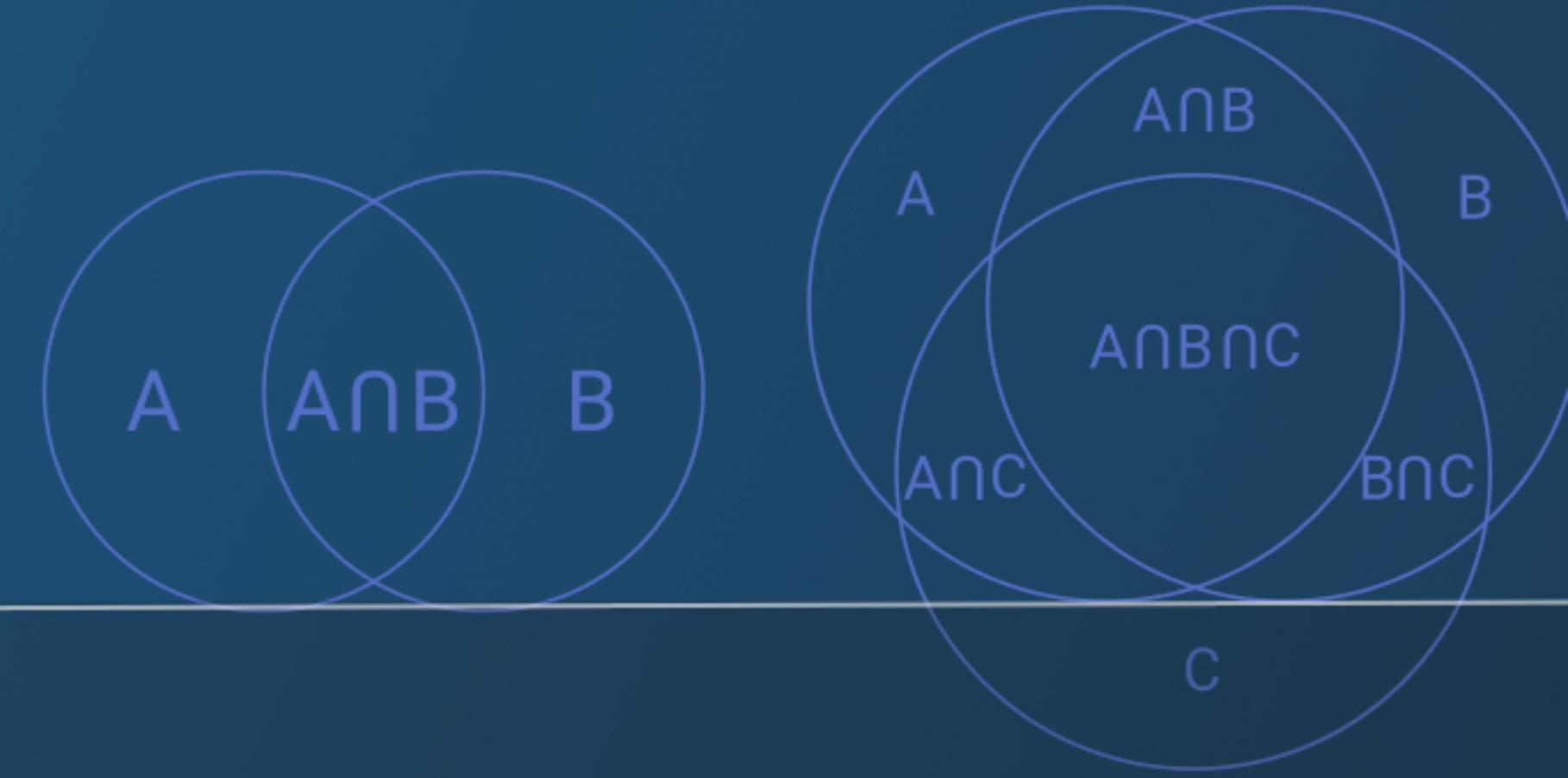
$$xxx(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} G(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} e^{-\frac{x^2}{2\sigma^2}}$$

Julia 程式語言學習馬拉松

Day 01



Cupay 陪跑專家 : James Huang



$$\begin{aligned}y'' &= \left[\ln(x + \sqrt{1 + x^2}) \right]' + \left(\frac{x}{\sqrt{1 + x^2}} \right)' \\&= \frac{1}{x + \sqrt{1 + x^2}} \left(\frac{\sqrt{1 + x^2} + x}{\sqrt{1 + x^2}} \right)' + \frac{1}{\sqrt{1 + x^2}} - \frac{x^2}{\sqrt{(1 + x^2)^3}} \\&= \frac{1}{\sqrt{1 + x^2}} + \frac{1}{\sqrt{1 + x^2}} - \frac{x^2}{\sqrt{(1 + x^2)^3}} \\&= \frac{2}{\sqrt{1 + x^2}} - \frac{x^2}{\sqrt{(1 + x^2)^3}} = \frac{2(1 + x^2) - x^2}{\sqrt{(1 + x^2)^3}} = \frac{2 + x^2}{\sqrt{(1 + x^2)^3}}\end{aligned}$$

Julia 安裝及簡介



重要知識點

- 認識 Julia 程式語言
- 安裝及建置 Julia 環境
- 開發及執行 Julia 的方式
 - Jupyter Notebook
 - REPL (Read-Eval-Print Loop) 環境
 - 從命令提示列 (command-line interface; CLI) 執行





Julia 的發展



- Julia 在 2009 年，由 Jeff Bezanson, Alan Edelman, Stefan Karpinski, Viral B.Shah 開始專案，並在 2012 年公開。
- 獲得 MIT Lincoln Laboratory, Intel, NASA... 等單位的贊助支持。
- 主要版本發佈：
 - v0.1: 2013 年 2 月
 - v0.6: 2017 年 6 月
 - v0.7: 2018 年 8 月
 - v1.0: 2018 年 8 月
- 目前最新穩定版本為 v1.3.1 (2019 年 12 月 30 日發佈)
 - 馬拉松使用的版本建議 1.2 以上



Julia 及 JuliaPro



VS



A Julia package crafted for your success

- Julia 是開源 (open source) 的，官方網站為 <https://julialang.org/>
- JuliaPro 是 Julia 作者們成立 Julia Computing 所推出的版本，官方網站為 <https://juliacomputing.com/>
- 不同的地方在於，JuliaPro 提供免費及付費版本，付費版本並提供企業技術支援服務，並內建安裝 JuliaPro 所支援的套件 (包含 Juno)，以及其他進階的功能。
- 馬拉松的課程內容，會以 open source 版本為準。



Julia 程式語言特色



特色

簡介

快速

透過 JIT (just-in-time) 的即時編譯，讓 Julia 的執行速度能比美 C/C++。

動態

如同 Python，Julia 也支援動態型別的開發，使用者可以自定資料類型。

多重分派 (Multiple Dispatch)

使用者可以定義同名函數的不同行為。

語法簡單

Julia 的設計目標之一就是希望語法能像 Python 或 Ruby 一樣簡單容易上手。

開源

Julia 是開放原始碼，並且有愈來愈多的第三方套件可以安裝使用。

與其他語言的整合

可與 Python, R, C... 等語言進行混合編程。



安裝及建置 Julia 環境



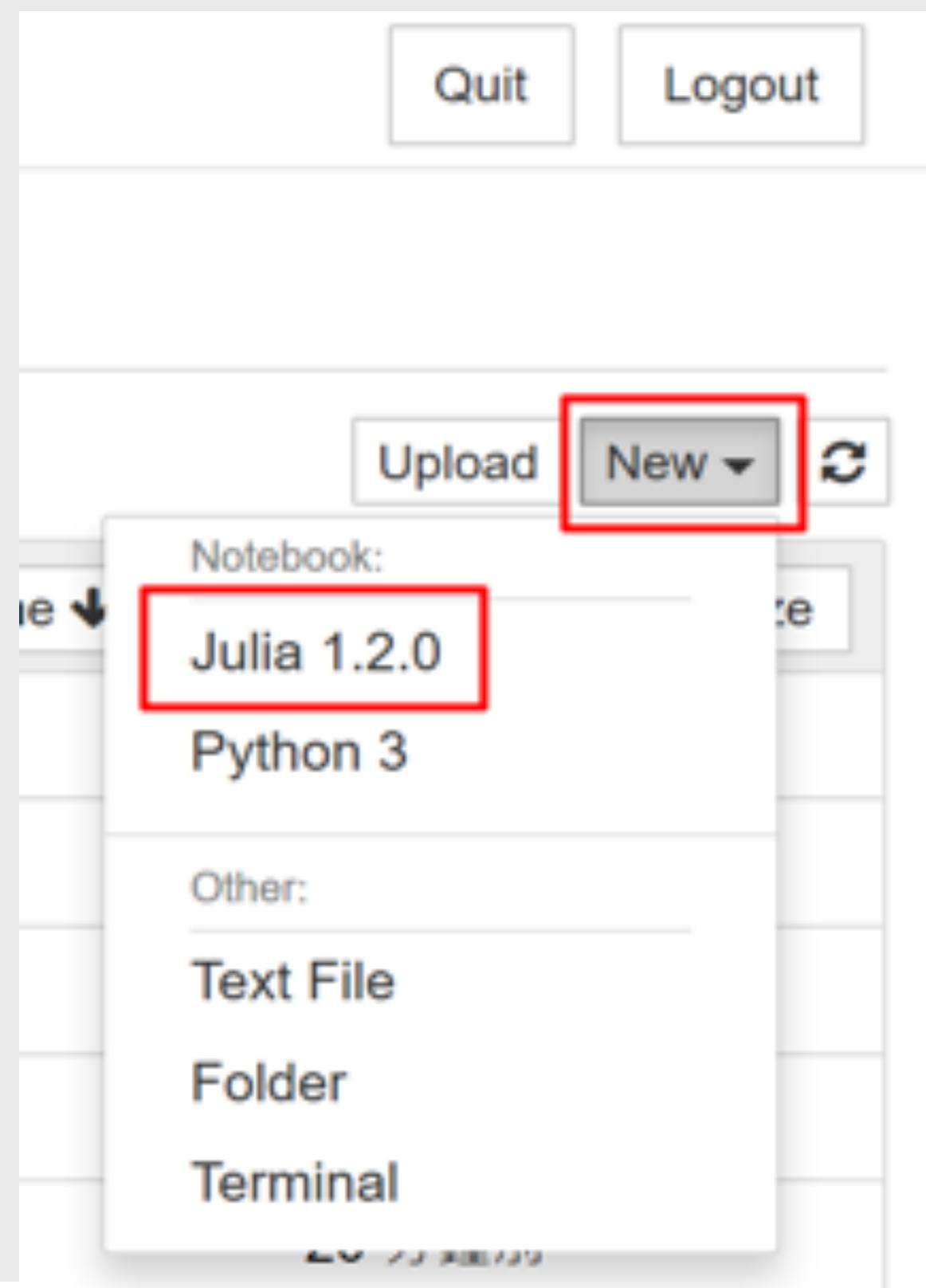
- 馬拉松會以 Windows 10 做為主要的範例環境，並在 Jupyter Notebook 進行示範及開發。若是在 macOS 或是 Linux 環境上有明顯差異之處，也會適時提醒。
- 安裝步驟請詳見附件：馬拉松_Julia安裝文件。請確認 Jupyter Notebook, Julia, IJulia 均安裝成功。
- 在本機執行 Julia 程式的三種方式：
 - Jupyter Notebook
 - REPL (Read-Eval-Print Loop)
 - Command-Line Interface 命令列



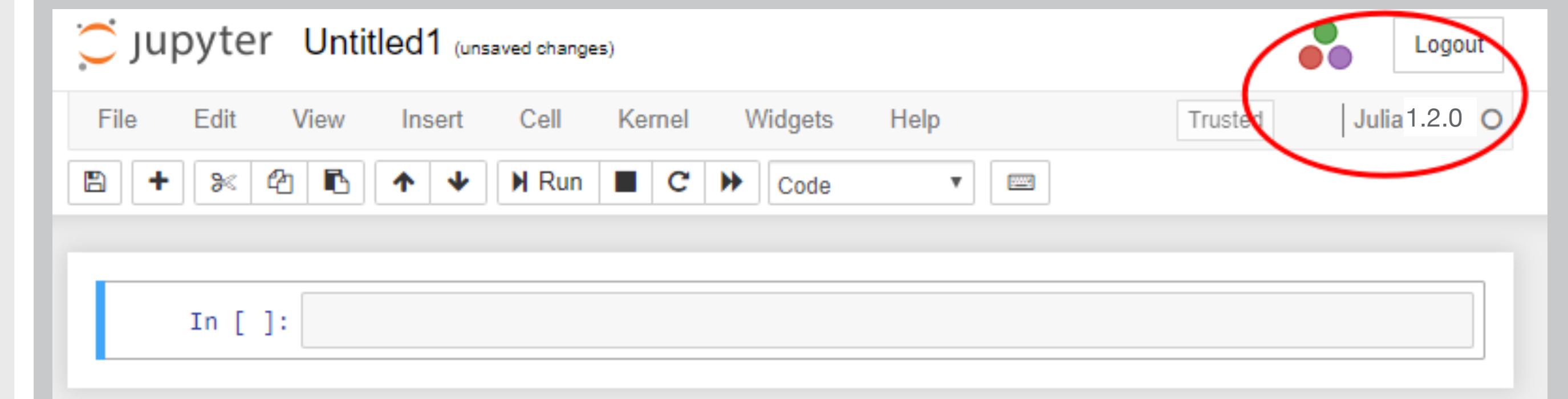
從 Jupyter Notebook 執行 Julia



- 開啟 Jupyter Notebook
- 新增一個新的 Julia 1.2.0 的 Notebook



- 新增成功後，可以看到右上角已顯示是 Julia 1.2.0 Kernel 的 Notebook





查看 Julia 版本及系統資訊



呼叫 `versioninfo()` 互動式工具函數，確認 IJulia 環境安裝成功，並顯示 Julia 版本以及系統資訊。



In [1]: `versioninfo()`

Julia Version 1.2.0

Commit c6da87ff4b (2019-08-20 00:03 UTC)

Platform Info:

OS: Windows (x86_64-w64-mingw32)

CPU: Intel(R) Core(TM) i7-4750HQ CPU @ 2.00GHz

WORD_SIZE: 64

LIBM: libopenlibm

LLVM: libLLVM-6.0.1 (ORCJIT, haswell)

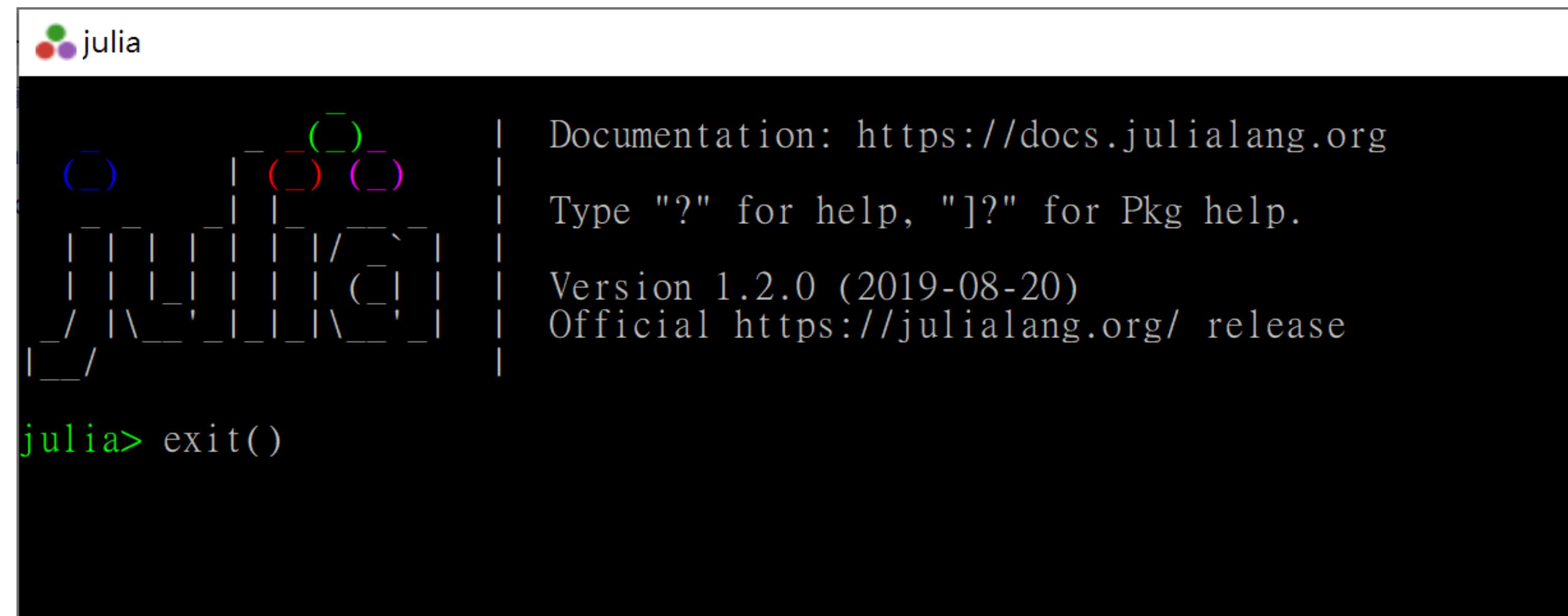
In []:



從 REPL (Read-Eval-Print Loop) 執行 Julia



- REPL 是互動式的命令執行環境。
- 從安裝 Julia 時建立的捷徑，啟動 REPL 模式
 - Windows 10: [開始]選單 ---> Julia 1.2.0 ---> Julia-1.2.0
 - macOS: 應用程式 ---> Julia-1.2.0



```
julia
julia> Documentation: https://docs.julialang.org
julia> Type "?" for help, "]?" for Pkg help.
julia> Version 1.2.0 (2019-08-20)
julia> Official https://julialang.org/ release
julia> exit()
```



命令提示列執行 Julia



如果已將 Julia 程式編寫存成 .jl 檔，也可以透過命令提示列執行。



```
C:\source>julia hello_julia.jl
hello julia
C:\source>
```

知識點 回顧

- 安裝及建置 Julia 環境
- 開發執行 Julia 程式的三種不同方式：
 - 1 · 在 Jupyter Notebook 的互動環境中
 - 2 · 從 REPL 的互動命令環境
 - 3 · 在文字編輯器或其他 IDE 中編寫，透過命令列指令執行
- 更詳細的安裝及建置步驟，請參考今天所附的 Julia 安裝文件。



推薦閱讀

- 安裝及建置 Julia 開發環境實戰教學
- 深入淺出 Julia 套件管理 (Package Management)
- 有關於 REPL 和 Command-Line 執行的入門介紹：
Julia 官方文件 Getting Started
- 在馬拉松活動中開發環境以 Jupyter Notebook 為主，其他的開發環境請自行參考，例如 Juno 是一個受歡迎的 Julia 程式語言開發環境，熟悉 Atom IDE 的話可以自行參考 <https://junolab.org/>。





解題時間

請跳出 PDF 至官網 Sample Code
& 作業開始解題