

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

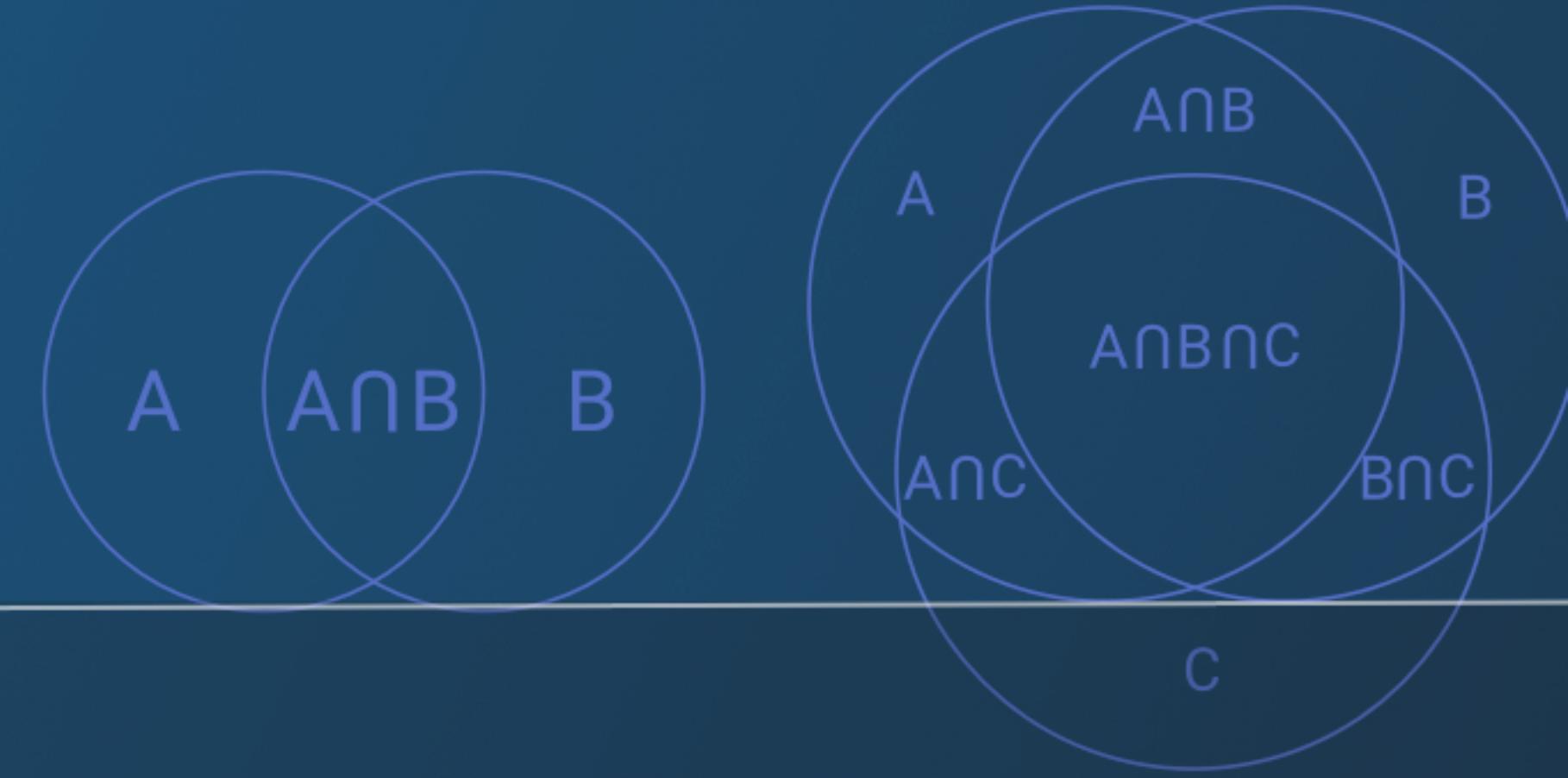
$$x(X, O) = -\frac{x}{\sigma^2} G(X, O) = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

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

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

Julia 程式語言學習馬拉松

Day 12



$$\ln(x + \sqrt{1 + x^2}) + x - \frac{1}{x + \sqrt{1 + x^2}} \left(1 + \frac{x}{\sqrt{1 + x^2}} \right)$$



cupay 陪跑專家 : James Huang

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

型別系統簡介 (Type System)





重要知識點

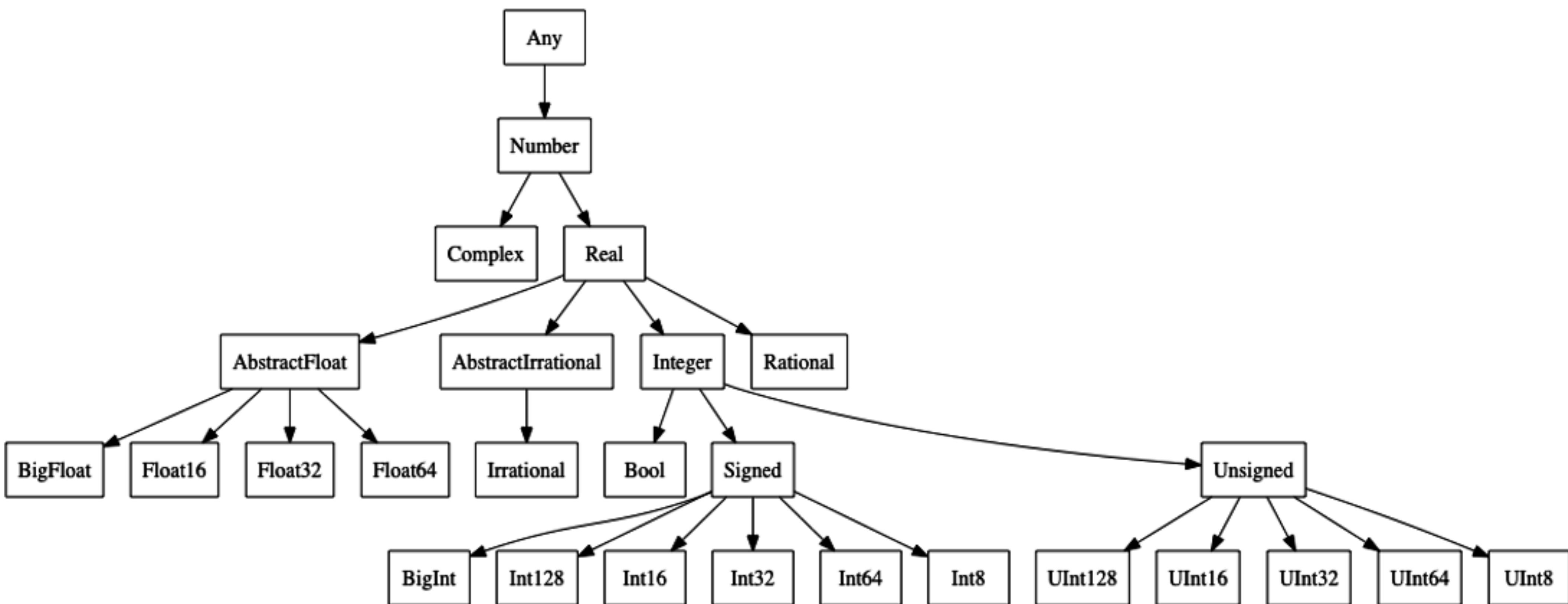
- 在 Julia 語言中擁有龐大且複雜的型別 (Type) 架構，可以賦予數值、函式、表達式、型別本身特定的型別。在今天的內容中，將針對不同的型別定義及用法進行說明：
 - 抽象型別 (Abstract Types)
 - 原始型別 (Primitive Types)
 - 複合型別 (Composite Types)
- Julia 的型別系統中有幾個特點：
 - 所有“值 (value)”均為物件，而物件必有其從屬的型別
 - 型別屬於 run-time type，值唯有在執行時才確定其型別。
 - 變數是沒有型別的。





型別階層範例 – 數值

- 以數值相關的型別為例，其最上層父型別為 Any (Any 也是所有型別的父型別)。





型別 (Type) 的宣告

- 型別的宣告是使用 :: 運算子，可以針對變數、表示式 (expression)、與函式進行型別的宣告。

類型	範例語法	說明
宣告變數型別	local x::Int8 = 10	僅能針對 local 變數，若宣告 global 變數的型別時會產生錯誤。
表示式宣告型別	(1+2)::Int64	若型別不符合會產生 TypeError。
函式宣告型別	function foo()::Float64 return 2 / 2 end	若宣告函式回傳型別的，回傳值會進行轉換為指定的型別，若無法轉換的話則會產生錯誤。



抽象型別 (Abstract Types)



- 抽象型別有幾個特性：
 - 無法被實例化 (Instantiate)；若對抽象型別進行實例化，Julia 會自動找到符合可實例化的原始型別。
 - 抽象是可以延伸的，可以有多層次
 - 可以運用抽象型別來撰寫泛用 (generic) 函式，以做為函式的基本行為 (behavior)
- 抽象型別的最上層是 Any，也是所有型別的最上層父型別；抽象型別的最下層則是 Union{}，也是所有型別的子型別。
- 抽象型別的宣告可使用 <: 運算子指定其父型別。如果沒有指定的話，則父型別為 Any。



原始型別 (Primitive Types)



- 原始型別可以被實例化 (Instantiate) 產生物件，我們常用的 Int64、Float64... 等均為原始型別。
- 原始型別宣告時可使用 `<:` 運算子指定其父型別。如果沒有指定的話，則父型別為 Any。
- 原始型別宣告時可以指定型別需要的儲存空間，以 Int64 的宣告為例：primitive type Int64 <: Signed 64 end



複合型別 (Composite Types)



- 複合型別是用關鍵字 `struct` 來進行宣告，成員可以指定型別或不指定，不指定即為 `Any` 型別。
- 複合型別中的成員型別也可以是另一個複合型別。
- `struct` 可分為可變 (`mutable`) 與不可變的，其差異在於在實例化 (`instantiate`) 之後 `mutable struct` 的成員值是可以被改變的，而不可變複合型別成員值不可以被改變，嘗試改變時會產生錯誤。



型別聯合 (Union)



- 型別聯合是透過 Union 的方式，可以讓物件的型別限定在數個指定的型別之內，但又不需要使用更高層級或是抽象型別來宣告，例如：如果希望型別可能是 Int64、Int32、或是 Float64 時，可以透過 Union 來達到。
 - 範例：`Int64OrInt32OrFloat64 = Union{Int64, Float64, Int32}`
- Union 可以包含不同父子關係的型別。



型別參數化 (Parametric Types)



- 型別參數化可以彈性地型別，而不影響程式的彈性及重覆使用性，也能降低程式的複雜度。
- 複合型別參數化
 - 範例：

```
1 struct Point{T}
2     x::T
3 end
```
- 抽象型別參數化
 - 範例：

```
1 abstract type Pointy{T} end
```
- 完整的型別參數化範例請參考今日程式碼。



型別 (Type) 常用函式



函式	範例語法	說明
supertype()	supertype(Real)	查看上一層父型別
subtypes()	subtypes(Number)	查看所有子型別
isa()	isa(a, b)	判斷型別是否相同
dump()	dump(a)	查看型別的內部結構
typeof()	typeof(a)	查看型別
fieldnames()	fieldnames(a)	查看複合型別成員

知識黑點 回顧

- Julia 的型別系統為動態 (dynamic) 型別系統，但也提供類似靜態 (static) 型別系統的優點。
- 原始型別不會是彼此的子型別，而且會是最下層的子型別；抽象型別可以是原始型別的父類別。
- 了解型別的相關操作、階層關係、型別參數化，有助於更有彈性地使用型別，增進重覆使用性，或是建立自定型別。
- Julia 型別系統的架構提供了多重分派 (multiple dispatch) 的基礎，有關於多重分派將會在之後的內容中介紹。



推薦閱讀

- [Julia 型別官方文件](#)
- [Julia系列教程6--類型](#)





解題時間

請跳出 PDF 至官網 Sample Code
& 作業開始解題