

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

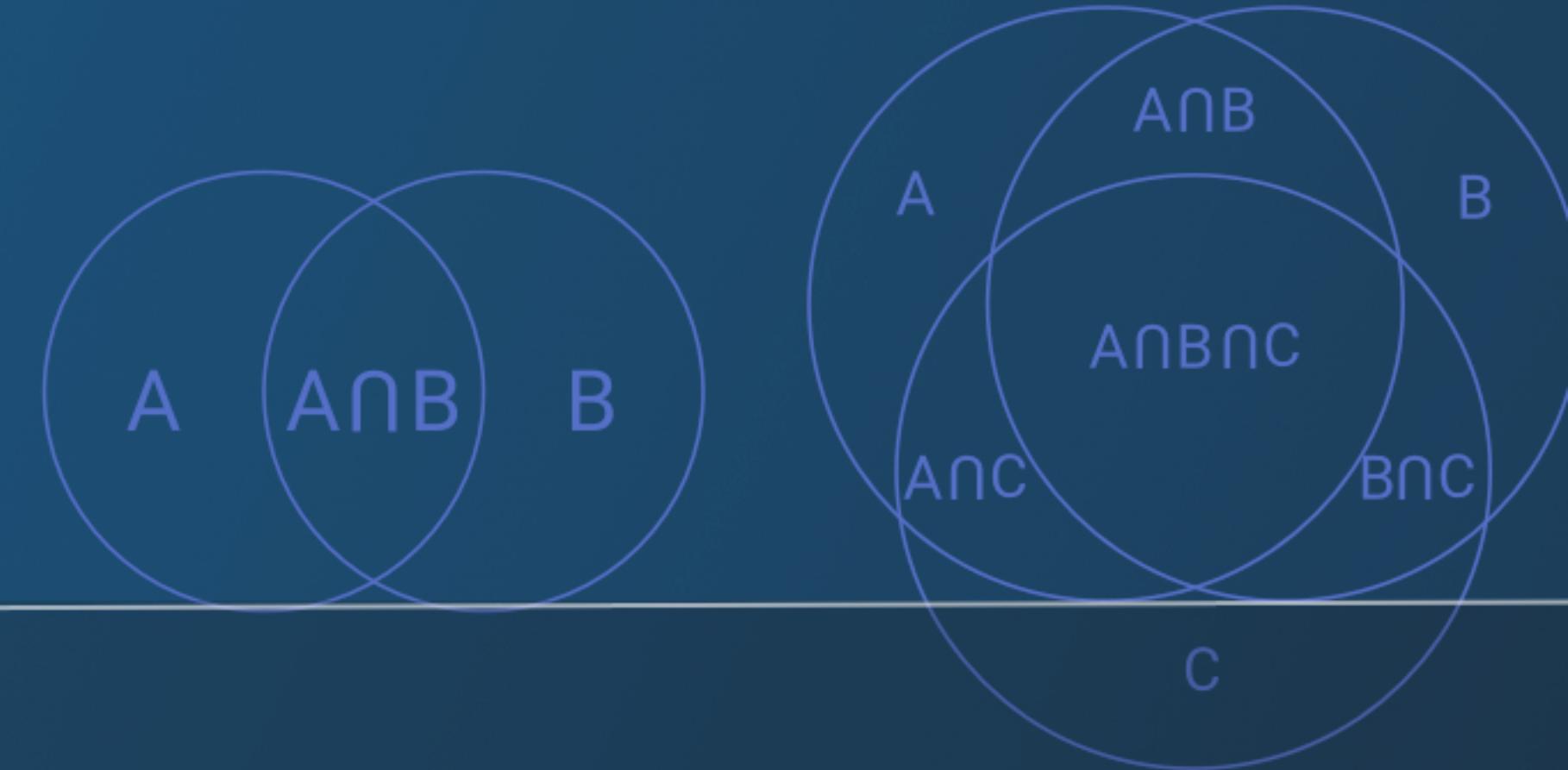
$$x(X, O) = -\frac{x}{\sigma^2} G(X, O) = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

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

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

Julia 程式語言學習馬拉松

Day 9



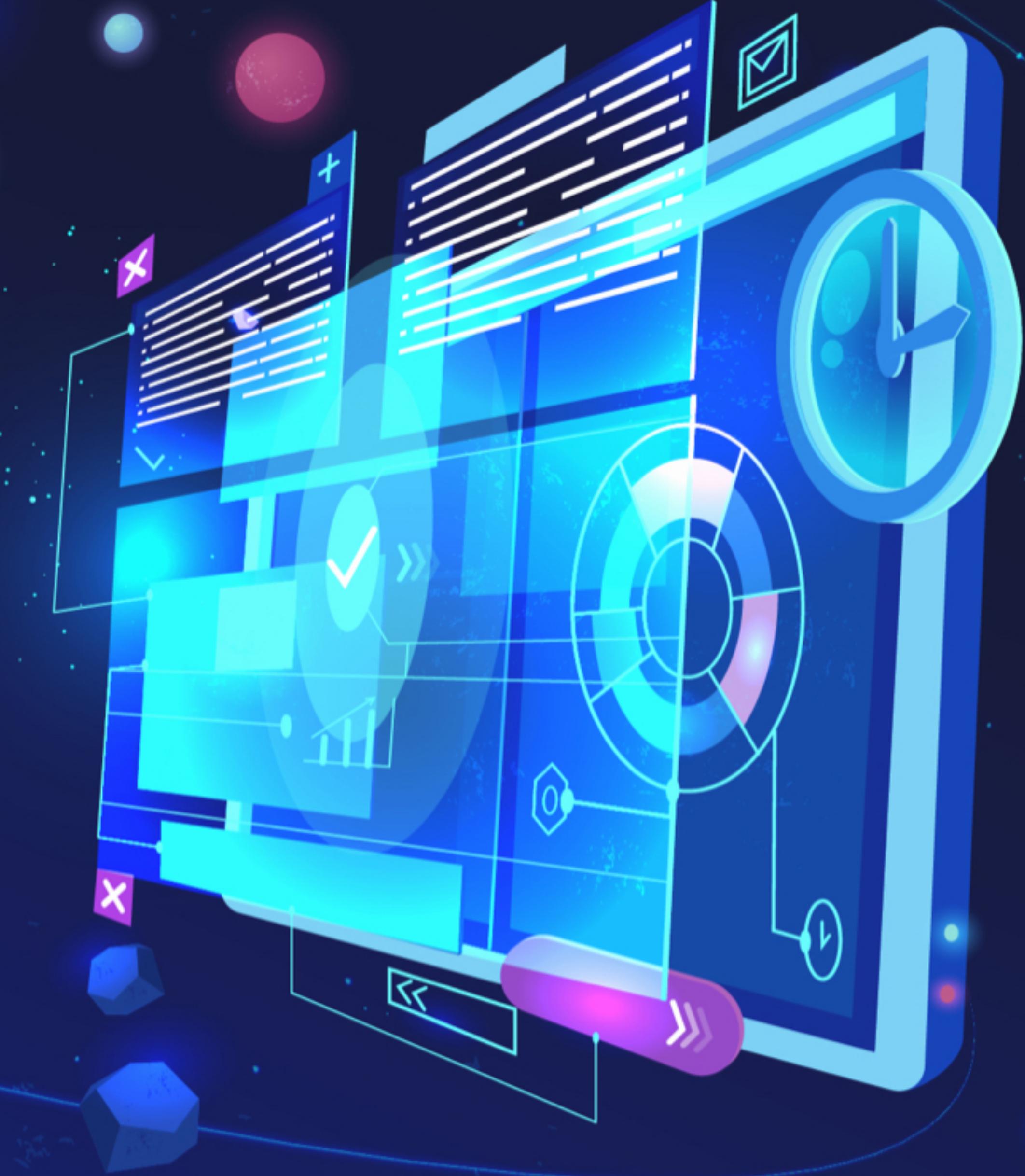
$$\ln(x + \sqrt{1+x^2}) + x - \frac{1}{x + \sqrt{1+x^2}} \left(1 + \frac{x}{\sqrt{1+x^2}} \right)$$



cupay 陪跑專家 : James Huang

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

陣列 (Array)





重要知識點



- Julia 的陣列 (array) 資料結構以及其操作。
- 在多維陣列提到索引或元素的位置時，我們將採用”直”行 (column) ”橫”列(row) 的翻譯來表示 (請注意，這樣的翻譯與大陸剛好相反)。



陣列的建立

- 最簡單的 array 創建方式：用中括號包含所有陣列元素 (element)

```
In [1]: 1 A = [1 5.0 3]
```

```
Out[1]: 1×3 Array{Float64,2}:
 1.0 5.0 3.0
```

- 用分號可以創建二維陣列：

```
In [1]: 1 A = [1 2; 3 4; 5 6]
```

```
Out[1]: 3×2 Array{Int64,2}:
 1 2
 3 4
 5 6
```



陣列的建立

- 通常一維陣列也稱為向量 (vector)，二維陣列稱為矩陣 (matrix)。
- 實際上列陣列在 Julia 中也是以二維維度存在。

```
In [4]: 1 A = [1 5.0 3]
```

```
Out[4]: 1×3 Array{Float64,2}:
 1.0 5.0 3.0
```

```
In [5]: 1 # ndims 函數回傳陣列的維度數
2 ndims(A)
```

```
Out[5]: 2
```

- 雖然在創建行陣列 (或稱行向量) 時可以用逗號或分號換列，但是在創建二維陣列 (矩陣) 時則不能用逗號換列，所以建議可以統一用分號，就不會混淆了。



陣列的建立

- 創建陣列時，元素可以是不同的型別，例如：[1.0, “a string”, 3]
- 創建陣列時，可以讓 Julia 自動識別元素型別而不指定。
- 要指定型別的話，在陣列前面加上型別即可，但要注意如果指定的型別無法被精確轉換時(例如欲將浮點轉為整數)，系統會拋出 error。

```
In [18]: 1 Float32[1, 2, 3, 4]
```

```
Out[18]: 4-element Array{Float32,1}:
 1.0
 2.0
 3.0
 4.0
```

```
In [19]: 1 Int32[1.1 2 3]
```

```
InexactError: Int32(1.1)
```



陣列的建立

- 可以透過下列常用函式創建並初始化陣列中的元素。
- 請參照今日範例示範下列函式的操作。

函式	描述
zeros(T, dims...)	陣列元素初始化為 0
ones(T, dims...)	陣列元素初始化為 1
trues(dims...)	陣列元素初始化為 true 的 BitArray 類型
falses(dims...)	陣列元素初始化為 false 的 BitArray 類型
rand(T, dims...)	產生元素為隨機介於 [0, 1 區間的均勻分布數字
randn(T, dims...)	產生元素為隨機常態分布的數字



陣列的建立

- 也可以使用 collect() 函式來建立陣列。
- 搭配 reshape() 函式，建立多維陣列。

```
In [22]: 1 # 建立一維陣列，起始值為 1，數值間隔 2，結束值 10  
         2 collect(StepRange(1, Int8(2), 10))
```

```
Out[22]: 5-element Array{Int64,1}:  
          1  
          3  
          5  
          7  
          9
```

```
In [23]: 1 # 建立 5x2 陣列  
         2 reshape(collect(1:1:10), (5, 2))
```

```
Out[23]: 5×2 Array{Int64,2}:  
          1   6  
          2   7  
          3   8  
          4   9  
          5  10
```



陣列的索引

- 透過索引取得陣列元素值，有線性索引和笛卡兒索引兩種方式。
- Julia 的索引值起始是 1 而非 0，和多數程式語言不同。

```
In [24]: 1 A = [1 2 3; 4 5 6; 7 8 9; 10 11 12]
```

```
Out[24]: 4×3 Array{Int64,2}:
 1   2   3
 4   5   6
 7   8   9
10  11  12
```

```
In [25]: 1 # 線性索引，得到 A[3, 2] 的值
2 A[7]
```

```
Out[25]: 8
```

```
In [26]: 1 # 笛卡兒索引
2 A[3, 2]
```

```
Out[26]: 8
```



陣列的遍歷(Traversal)

- 下列範例是搭配迴圈及索引，列出陣列中所有元素。

In [28]:

```
1 # 使用笛卡兒索引將陣列中所有元素印出
2 A = [1 2 3; 4 5 6; 7 8 9; 10 11 12]
3
4 for i = 1:size(A, 1), j = 1:size(A, 2)
5     print(A[i, j], " ")
6 end
```

```
1 2 3 4 5 6 7 8 9 10 11 12
```

In [29]:

```
1 # 使用線性索引將陣列中所有元素印出
2 for i = 1:lastindex(A)
3     print(A[i], " ")
4 end
```

```
1 4 7 10 2 5 8 11 3 6 9 12
```

In [30]:

```
1 # 用迭代的方式將陣列中所有元素印出
2 # 也可以用 in 取代 ∈
3 for x ∈ A
4     print(x, " ")
5 end
```

```
1 4 7 10 2 5 8 11 3 6 9 12
```



加入或剔除陣列的元素



- 加入元素到陣列中，
 - `push!()` 函式，加入元素到陣列的尾端。
 - `pushfirst!()` 函式，加入元素到陣列成為第一個元素。
- 剔除陣列的元素，
 - `pop!()` 函式，將陣列的最後一個元素剔除。
 - `popfirst!()` 函式，將陣列的第一個元素剔除。



陣列的組合



- 若陣列中含有陣列 (或稱巢狀陣列) , Julia 會自動依維度展開。
- 但是如果巢狀內外陣列維度不相同，則會拋出 error。

```
In [20]: 1 [1 2 [3 4]]
```

```
Out[20]: 1×4 Array{Int64,2}:
 1 2 3 4
```

```
In [21]: 1 # 巢狀內外陣列維度不相同
 2 [1; 2 [3; 4]]
```

```
DimensionMismatch("mismatch in dimension 1 (expected 1 got 2)")
```



陣列的組合

- 在前一頁的巢狀陣列組合的背後，其實是呼叫了 `cat()`、`vcat()`、`hcat()`、或 `hvcat()` 函式。
請參考今天範例程式，可以更清楚了解各個函式的功用。

函式	描述
<code>cat(A1, A2, A3, dims=k)</code>	依維度k進行合併
<code>hcat(A1, A2, A3)</code>	依行合併陣列
<code>vcat(A1, A2, A3)</code>	依列合併陣列
<code>hvcat(k, a, b, c)</code>	依照指定每列的行數 k 合併數值成為陣列



陣列的切片(Slicing)和視圖(View)



- 陣列切片，也就是將陣列中的子陣列擷取出來，可以透過幾個方式：
 - 使用索引，例如： $A[1, 2]$
 - 使用範圍，例如： $A[2, 2:end]$
 - 使用條件式，例如： $A[A .> 8]$ 。請留意使用條件式切片回傳的將會是一維陣列。
- 使用 `view()` 函式，可以產生陣列的視圖 (view)，例如：`view(A, 2:3, 2:3)`，就是將 A 陣列的子陣列擷取出來產生視圖，其型別為 SubArray。
 - 視圖也可以透過跟陣列一樣的方式進行操作，但是特別要留意的是，視圖中的元素值被改變時，原始陣列對應的元素值也會被改變。



陣列排序函式



- Julia 提供的常見排序函式有下列幾個，預設的排序方法為 QuickSort，也是 sort() 函式使用的方法。
 - Sort()
 - QuickSort()
 - InsertionSort()
 - MergeSort()
 - PartialQuickSort()
- 針對常見的排序函式使用，請參照今日範例程式。



陣列 - 搜尋



- 搜尋某值是否存在於陣列中，可以用 `in`，例如：`in(3, [1, 3, 5, 7, 9])`，回傳值為 `true` 或 `false`。
- 使用 `findall()` 函式，搜尋並回傳結果值的索引，例如：`findall(in(1), [5, 3, 2, 1])` 會回傳索引值 `4`。
- 使用 `findmax()` 及 `findmin()` 函式，回傳找到的值及其索引值，例如：`findmax([2, 3, 1, 5])` 會回傳 `(5, 4)`，即最大值為 `5`，其索引位值為 `4`。



陣列 - 搜尋



- 若是陣列已排序，可以使用 `searchsorted()`, `searchsortedfirst()`, `searchsortedlast()`，取得該值的索引值，特別注意的是 `searchsorted()` 回傳的是索引值範圍 (range)。



陣列常用函式



函式

eltype(A)

length(A)

ndims(A)

size(A)

getindex(A, 2, 2)

eachindex(A)

lastindex(A)

axes(A)

strides(A)

描述

陣列A中的元素類型。

陣列A中的元素總數，例如 3×2 的陣列其元素總數為 6。

陣列A的維度數，例如二維陣列維度數為2。

陣列A的維度，回傳值是 tuple 類型，例如 3×2 的陣列其回傳值為 $(3, 2)$ 。

回傳位於 $A[2, 2]$ 索引的值

陣列A所有索引值。

陣列A最後一個索引值。

陣列A所有維度的有效索引值，回傳值是 tuple 類型。

陣列A各維度在記憶體中的距離(元素數目)，回傳值是 tuple 類型。

知識點 回顧

- 今天介紹了常用的集合 (Collection) 型別：陣列，以及其操作。
- 陣列是最重要的集合型別之一，所提供的函式與操作相當多，建議在閱讀時，參考對照提供的範例程式。



推薦閱讀

- [官方文件 Multi-Dimensional Arrays](#)
- [官方文件 Sorting and Related Functions](#)
- [\[演算法\] 排序演算法\(Sort Algorithm\)](#)





解題時間

請跳出 PDF 至官網 Sample Code
& 作業開始解題