

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

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

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

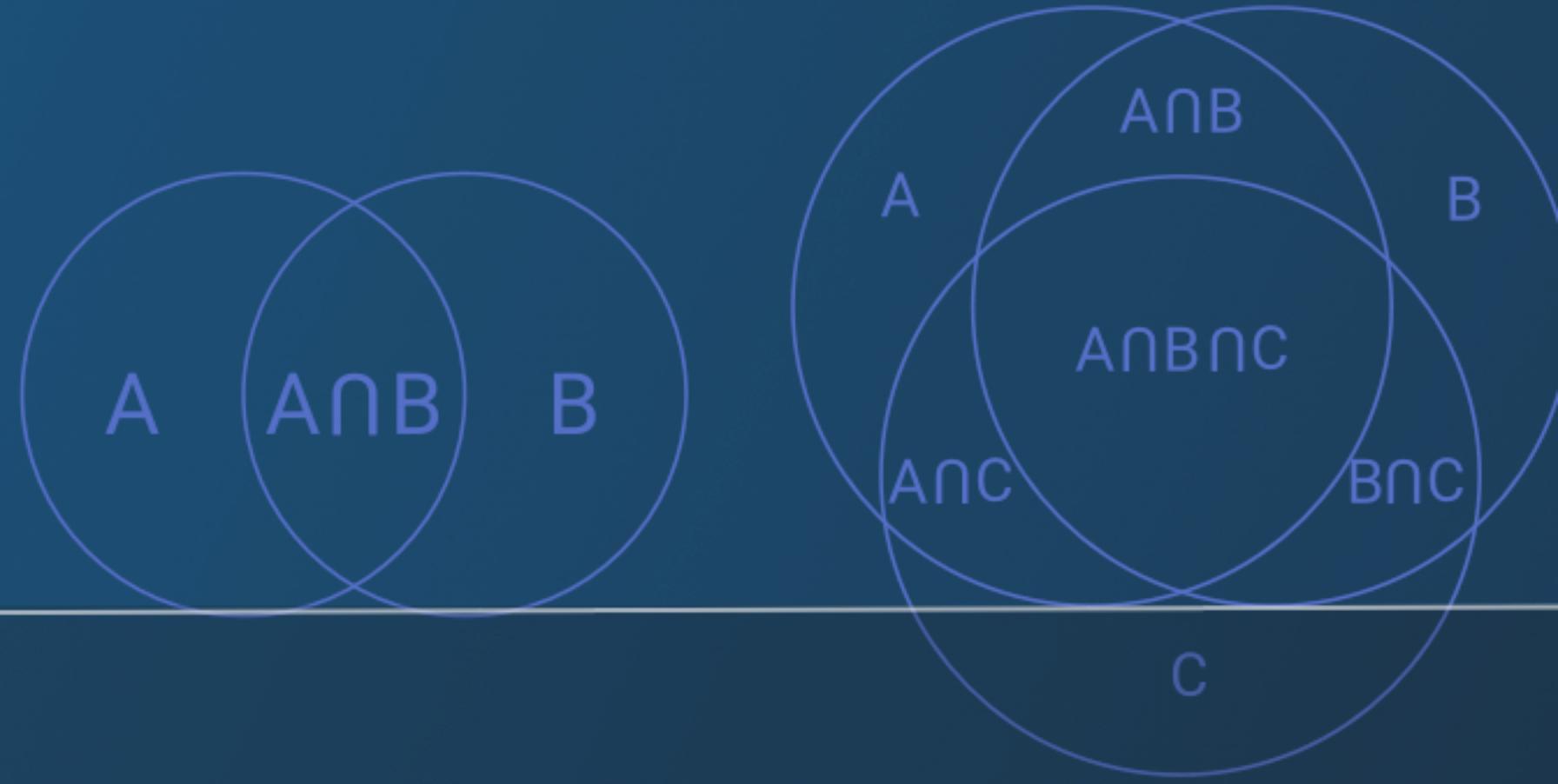
$$xxx(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} G(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} e^{-\frac{x^2}{2\sigma^2}}$$

Julia 程式語言學習馬拉松

Day 07



Cupay 陪跑專家 : James Huang



條件與迴圈





重要知識點



- 在程式語言的流程控制 (Control Flow) 中，條件和迴圈是常用的語法。
- 本日的內容將介紹 Julia Control Flow 下列相關語法：
 - if-elseif-else
 - while
 - for
 - 短路邏輯 (Short-circuit evaluation)
- 在 Control Flow 中的例外處理相關語法，會在後續的內容中詳細介紹。



比較運算子和邏輯運算子



- 在 Day 003 的時候介紹了運算子，其中比較運算子和邏輯運算子跟條件和迴圈的使用，關係非常密切。

比較運算子		邏輯運算子	
相等	<code>==</code>	<code>AND</code>	<code>&&</code>
不相等	<code>!=</code> <code>≠</code>	<code>OR</code>	<code> </code>
小於	<code><</code>	<code>NOT</code>	<code>!</code>
小於或等於	<code><=, ≤</code>		
大於	<code>></code>		
大於或等於	<code>>=, ≥</code>		



if-elseif-else 條件判斷



兩個條件判斷：

if 條件式

運算式

else

運算式

end

多個條件判斷：

if 條件式

運算式

elseif 條件式

運算式

else

運算式

end



if-elseif-else 條件判斷(續)



- if-elseif-else 條件式須以 end 關鍵字結尾。
- Julia 的 if-elseif-else 條件式中，比較特別的是，在 if / elseif / else 區塊中，本身就會 return value，這跟很多其他的程式語言不同。

```
1 function foo(x)
2     if x > 0
3         "positive!"
4     elseif x == 0
5         "zero"
6     else
7         "negative..."
8     end
9 end
10
11 println("x is ", foo(3))
```

x is positive!



if-elseif-else 條件判斷(續)



- if-elseif-else 條件式結果必須為 Bool (true 或 false)。在某些程式語言中常用 1 或 0 代表 true 或 false，但是在 Julia 中不允許，並會產生 error。

```
1 if 1
2     println("foo")
3 else
4     println("gee")
5 end
```

TypeError: non-boolean (Int64) used in boolean context

Stacktrace:

```
[1] top-level scope at In[8]:1
```



三元運算子 ?: (Ternary Operator)



- 三元運算子跟 if-elseif-else 的語法邏輯很相似。
- 三元運算子的運算式為 $a ? b : c$ ，其意義為：如果 a 條件式為 true 的話，就評估 b 運算式；若 a 為 false，則評估 c 運算式。
- 要注意是是在使用三元運算子的時候，? 和 : 前後與運算式都要有空格，否則會產生 error。



while 迴圈



語法：

while 條件式

運算式

end

- 跟 if-elseif-else 一樣，while 的條件式結果必須為 Bool (true 或 false)



for 迴圈

for 迴圈語法：

for 範圍

運算式

end

for 巢狀迴圈語法：

for 範圍1

for 範圍2

運算式

end

end

或是：

for 範圍1,範圍2

運算式

end



迴圈 break

- `break` 關鍵字在迴圈中，可以用來在符合條件情況下，中斷並跳出迴圈。下面左邊範例是 `while` 迴圈的例子，右邊是 `for` 迴圈的例子，都是在 `index` 值等於 5 的時候會跳出。

```
1 i = 1
2
3 while true
4   println(i)
5
6   if i >= 5
7     break
8 end
9
10 global i += 1
11 end
```

```
1 for j = 1:10
2   println(j)
3
4   if j >= 5
5     break
6   end
7 end
```



迴圈 continue



- `continue` 關鍵字在迴圈中，可以用來在某些條件情況下，直接執行下一個迴圈。下面左邊範例是 `while` 迴圈的例子，右邊是 `for` 迴圈的例子，都是在 `index` 值除以 3 餘數為 0 的時候印出值。

```
1 i = 0
2
3 while i < 10
4     i += 1
5     if i % 3 != 0
6         continue
7     end
8     println(i)
9 end
```

```
1 j = 1
2
3 for j = 1:10
4     if j % 3 != 0
5         continue
6     end
7     println(j)
8 end
```



短路邏輯 (Short-circuit evaluation)



- 當有多個條件式，可以透過邏輯運算子達到短路邏輯的結果。
 - && (AND)**: 兩條件式均為 true 的話就是 true
 - || (OR)**: 兩條件式至少一個 true 的話就是 true
 - 反之就是 false
- 第 3 個範例是 ! (反向) 的範例。

```
1 i = 1; j = 2
2
3 if i == 1 && j == 2
4     println("true")
5 else
6     println("false")
7 end
```

true

```
1 i = 1; j = 3
2
3 if i == 1 || j == 2
4     println("true")
5 else
6     println("false")
7 end
```

true

```
1 j = 2
2
3 if !(j == 2)
4     println("true")
5 else
6     println("false")
7 end
```

false

知識點 回顧

- 今天的內容介紹了
 - 條件判斷 if-elseif-else
 - while 迴圈
 - for 迴圈與巢狀 for 迴圈
 - 三元運算子
 - 比較運算子和邏輯運算子的運用，以及與條件與迴圈的結合，達到最小化求值。
- 條件式結果必須為 Bool (true 或 false)，與某些程式語言不同。
- 更多範例及說明，請參見本日範例程式和作業。



推薦閱讀

- Control Flow 官方文件





解題時間

請跳出 PDF 至官網 Sample Code
& 作業開始解題