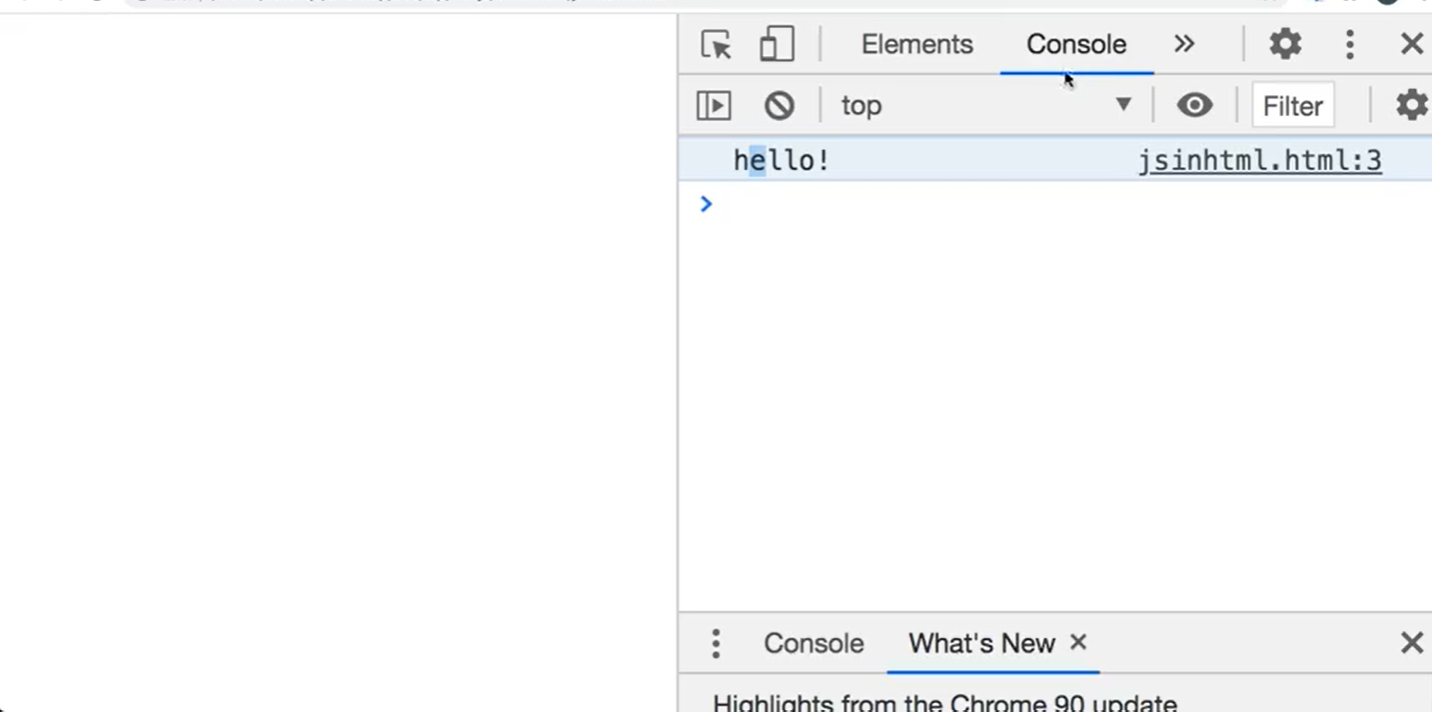


網頁設計要加入JavaScript的功能，要用<script>…</script>

如果要寫在網頁上就要加

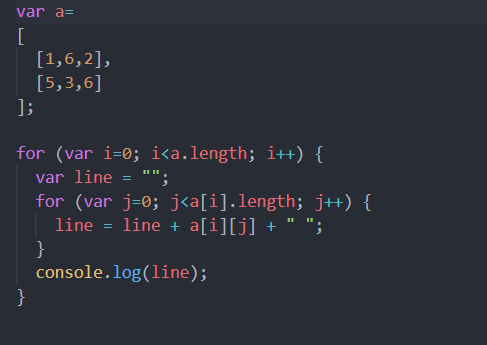


JavaScript可以嵌在HDML裡



Console.log只會出現在開發人員工具的Console裡，不會出現在網頁上

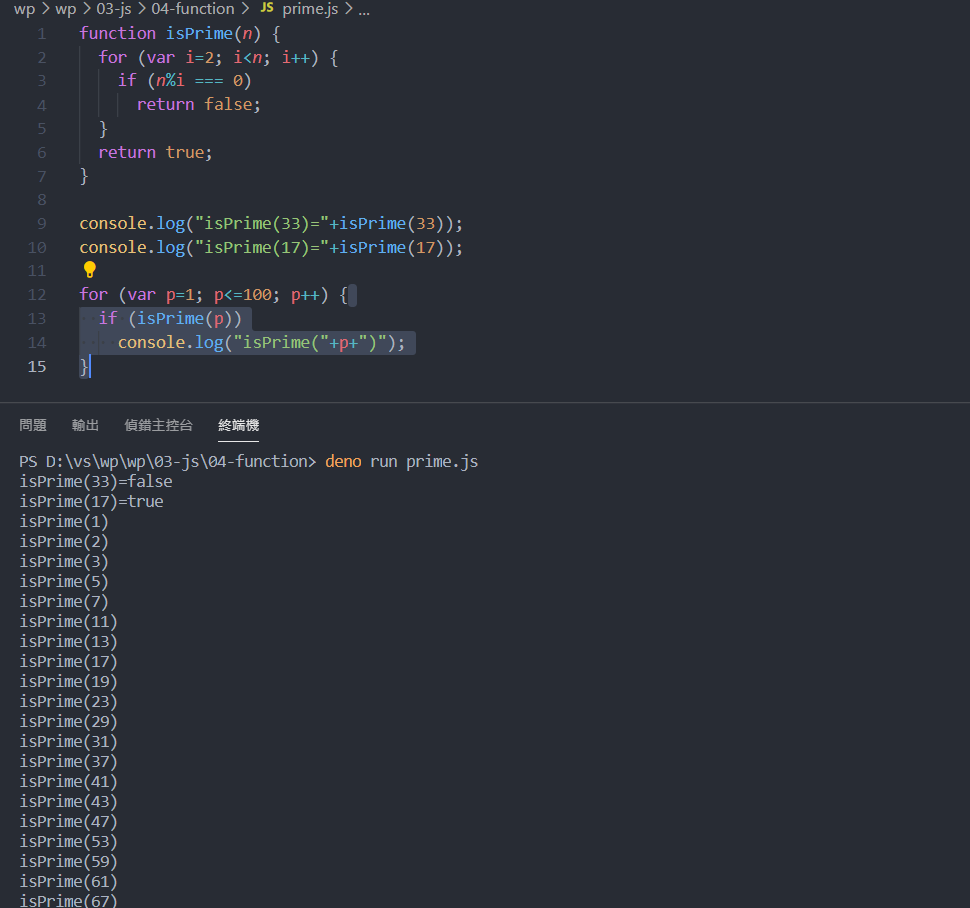
ARRAY



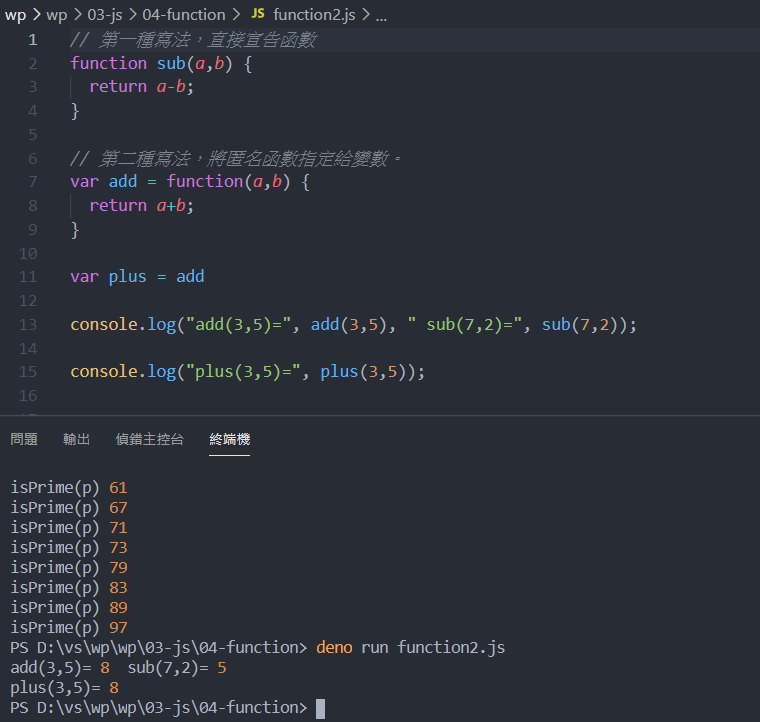
JavaScript的陣列用[]夸起來(C用{})，用a.length可以取得列數(2)

而a[0].length是取得列(0)裡面的字數(3)

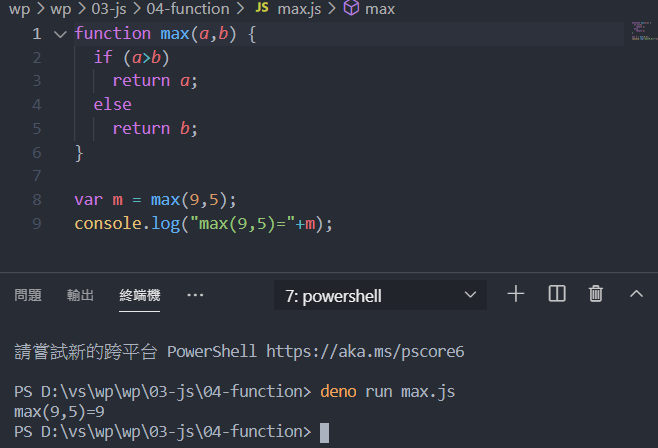
FUCNTION



“+p+”打印出p的值和夸號(+是代表加左右邊的符號)

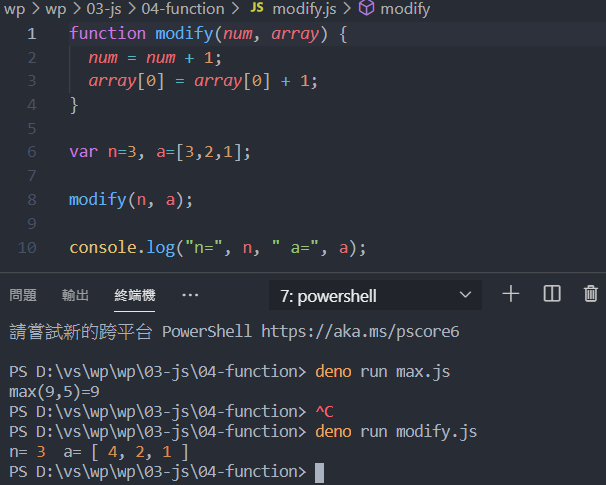


匿名函數把名字塞到funtion前面(add)



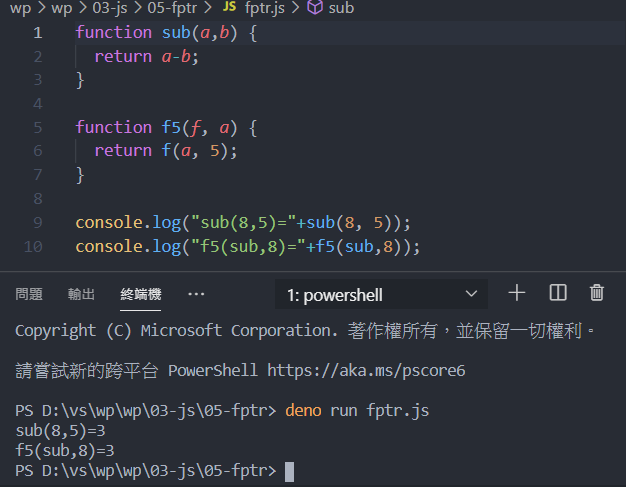
JavaScript的語法很隨性，一種問題可以用10種不同的寫法，而且分號可加可不加，像Phython，C就比較嚴謹。

逗號會有空格，而+m代表會緊跟在後面 (而且這裡的字還不會有顏色欸!)

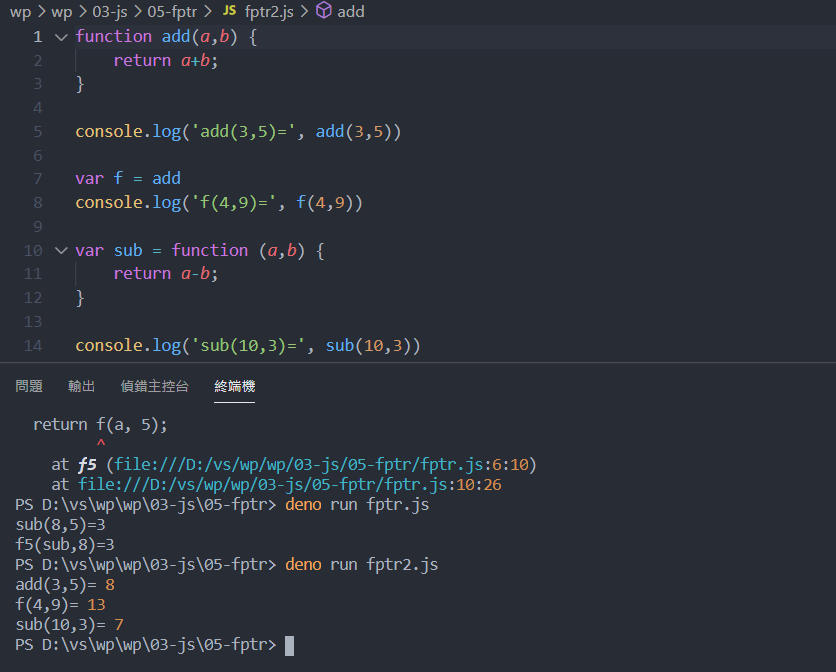


函數不會改變廣域變數的值，只是改到複製的值，原本的值不變。

而陣列就好像指標，複製位置，改的是位置裡的值。



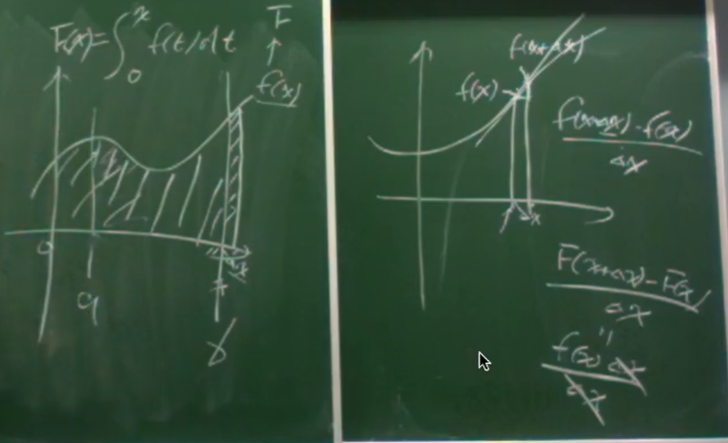
一個函數可以當參數，Funtion裡面的接收值可以接收函數，但Funtion要return函數型式，而且只能填函數



上面這個是用函數帶入參數，讓f===add函式

微分和積分為甚麼要放在一起??  


微積分基本定理 :　微分再積分就會等於他自己



上方的圖左半邊是積分，右半邊是微分

微分的基本公式 {f(x+Δx)-f(x)}/ Δx ，Δx代表很微小的距離，微分主要是求斜率

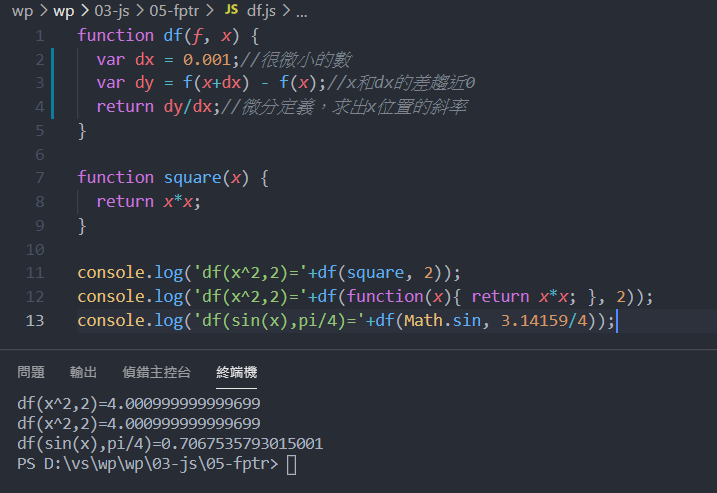
而積分是求面積，上圖左上方的F()代表0到x的面積，而我們再把這F帶入微分公式(右下)就會得到{F(x+Δx)-F(x)}/ Δx，也就是圖中x到Δx的面積

，會等於{f(x)\* Δx}/ Δx也就是f(x)。(”等於{f(x)\* Δx}/ Δx” 最後這個我不懂，為什麼要除Δx ?)

微積分很重要，處理連續函數。

而離散數學，處理分散數據。

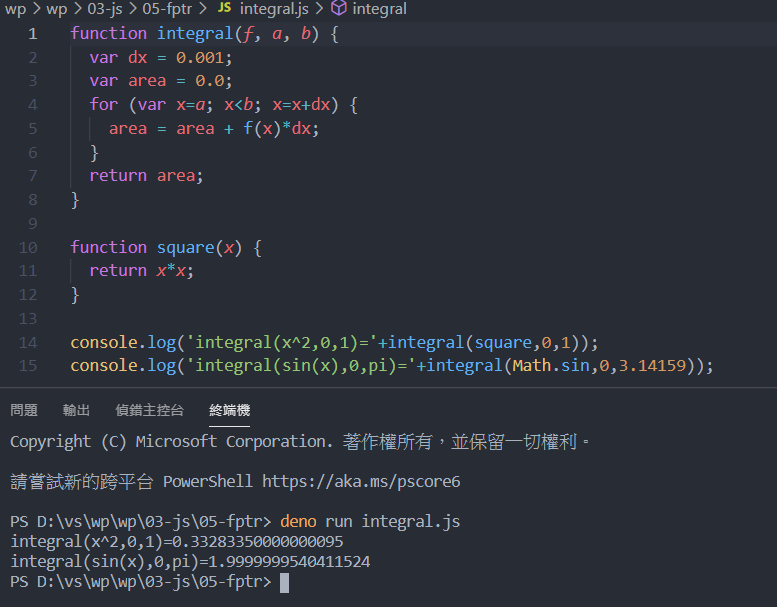
微積分函數



第一個輸出和第二個輸出都是一樣的，只不過一個是函式寫在外面，另一個是寫在裡面。

上圖的fucntion df()就是用微分定義下去寫的，其中，dx越小(通常0.001)越好

上面的funtion意思是===df(函式, 在x等於多少時函數微分值)



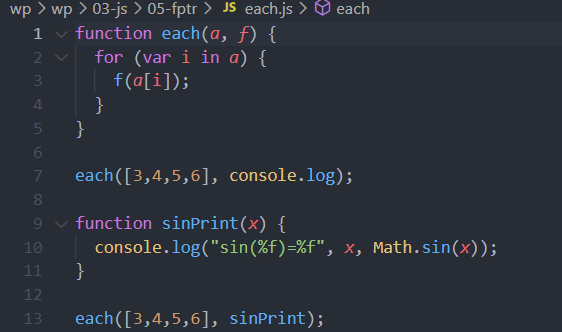
上圖也是用定義寫的，他把面積切成很多小塊再把他們加起來

一開始area是0，在x=a的地方慢慢加到x=b的地方

x慢慢累加，一次加dx，面積也一次加f(x)\*dx

下面是each(自定義funtion)的用法，較為進階

以函數為參數





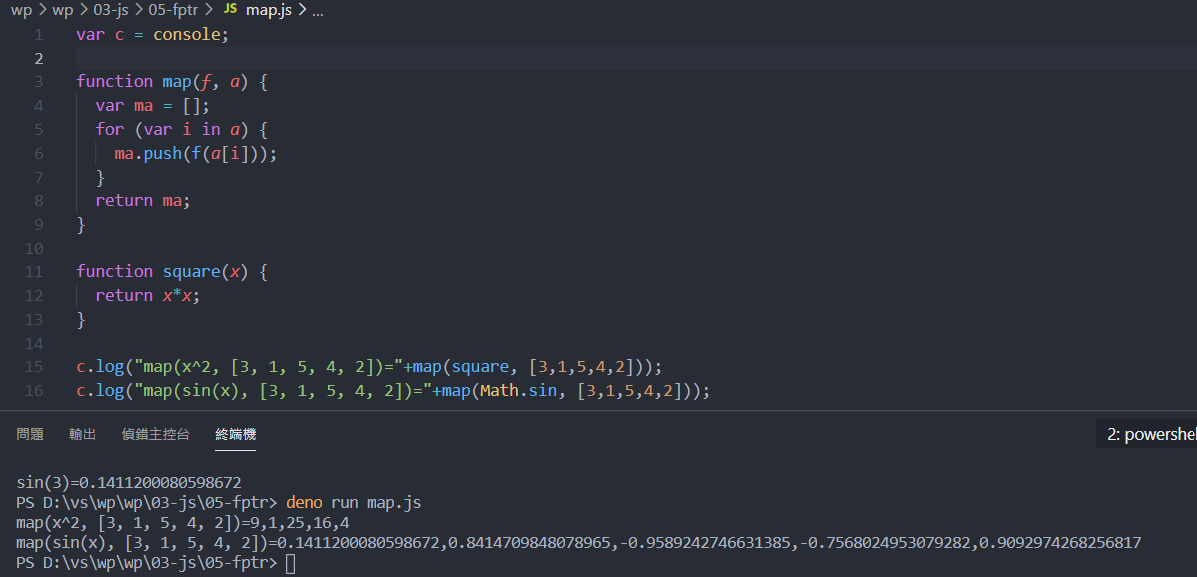
上面兩張圖的輸出是一樣的，差在一個是用in，一個是用of

**a的大小就代表他要跑幾次。如果是矩陣，有多少個元素就跑幾次**

**in會把變數(i)帶入索引值(0,1,2,3…)，a[i]代表a的第i個(=of)**

**of意思是把a裡面的元素(x)一個一個帶入函式**

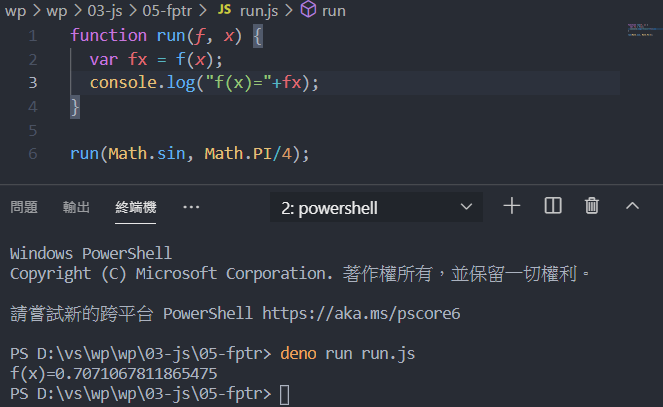
**用in和of下面的裡面的變數型態就是array( [] )**



map(function, array)，push是把a矩陣推到ma矩陣裡**(push把最後一位增加)**

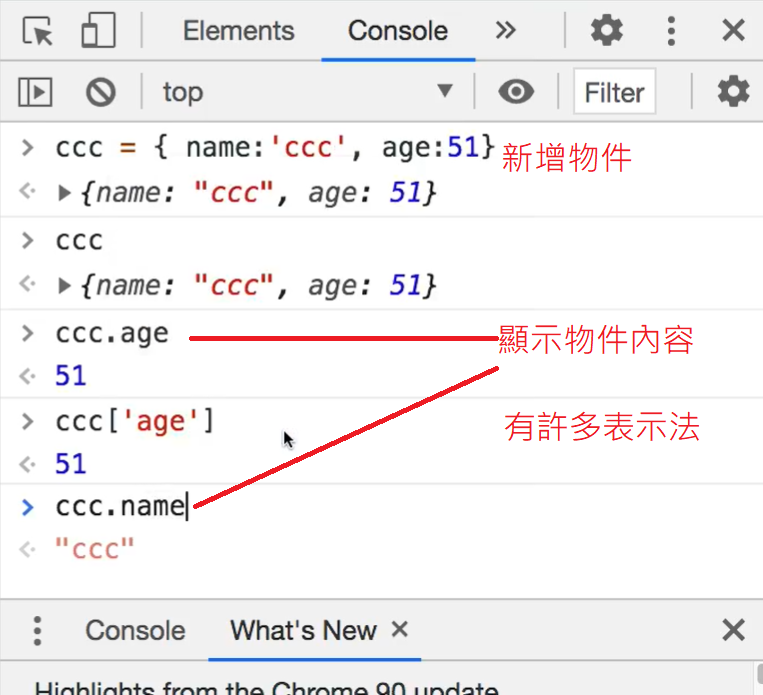
(用另一個存變化的值)

最後再回傳ma值，

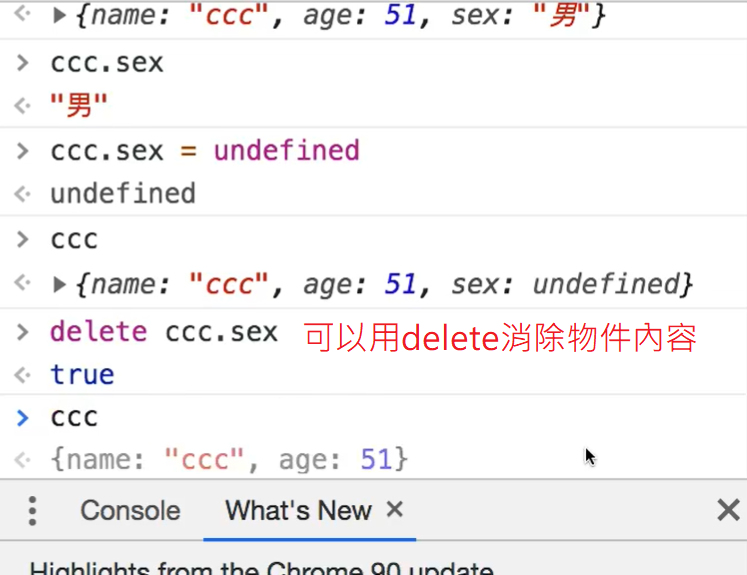


run其實沒做什麼，就只是把東西印出來

物件導向









可以透過新增矩陣，增加其他物件到另一個物件裡面

this是指自己函式加上.什麼，就是指物件裡的東西

Javascript在寫物件時可以用建構函數(constructor)(這個也算自定義)

EX:

class Circle {

constructor(r) {

this.r = r

}

area() {

return 3.14\*this.r\*this.r;

}

}

var c1 = new Circle(1)

console.log("c1.r=%d", c1.r);

console.log("c1.area()=%d", c1.area());

var c2 = new Circle(2)

console.log("c2.area()=%d", c2.area());

執行結果

csienqu-teacher:object csienqu$ node circleClass

c1.r=1

c1.area()=3.14

c2.area()=12.56

用constructor把變數丟到物件(this)裡面，要用時，直接呼叫var x = circle(?)

就可以把變數丟到函式裡面。

每次呼叫class裡的東西，要先宣告

\*\*\*重點!!!

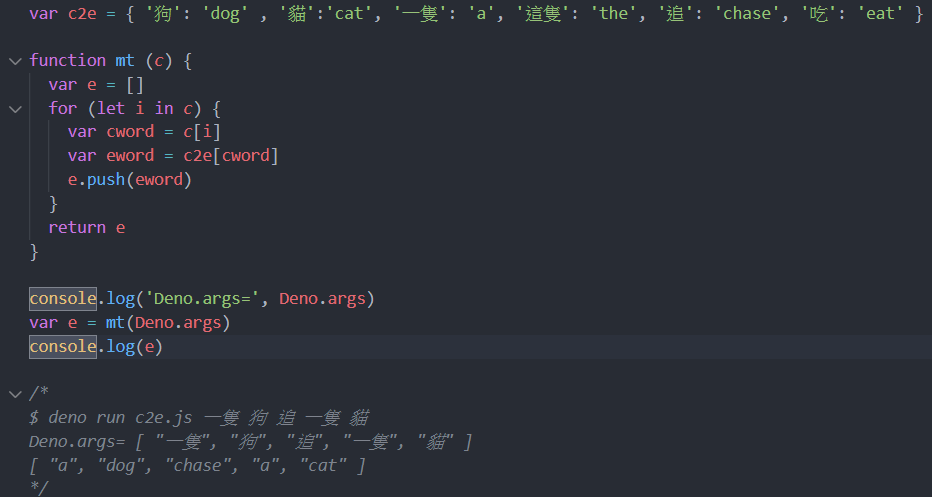
***MainGame/\*class名稱，可以視作一種種類(int之類的)\*/ mainGame/\*宣告名稱\*/***

***如果裡面有變數要存取(constructor)，就可以這樣宣告***

***MainGame mainGame = new MainGame(0);***

***也可以 MainGame = class MainGame(0)//這好像是古老式寫法，不確定***

***\*\*\****重點!!!

******

**Deno.args會偵測後面打的文字，之後用自訂函式mt帶入Deno.args**

**mt函式先把後面打的逐字存到c[]裡，再把這些字依序放入cword**

**然後由eword去存取c2e[cword]裡面的英文字//c2e.狗 = dog , c2e[‘狗’] = dog**

**之後再push到e[]裡然後回傳，就翻譯完成了(push把最後一位增加)**

**再用console.log印出，就完成了只能翻譯有限內容的程式**