```
print("j 陣列: ", terminator: " ")
for x in 0..<j.count {
    print("\(j[x]) ", terminator: " ")
}
print("")
```

(b)

```
var fruits = ["Orange", "Apple", "Banana", "Watermelon"]
var anotherFruits = fruits

print("fruits 陣列中有: ")
for names in fruits {
    print("\(names) ", terminator: " ")
}
print("")

print("anotherFruits 陣列中有: ")
for anotherNames in anotherFruits{
    print("\(anotherNames) ", terminator: " ")
}

fruits[0] = "Guava"
print("\n\n 改變 fruits 陣列的第一個元素為 Guava 後")
print("fruits 陣列中有: ")
for names in fruits {
    print("\(names) ", terminator: " ")
}
print("")

print("anotherfruits 陣列中有: ")
for anotherNames in anotherFruits{
    print("\n\(anotherNames) ", terminator: " ")
}
print("")
```

(c)

```
var arr = [1.1, 2.2, 3.3, 4.4, 5.5]
for i in 0..<arr.count {
    print("\(arr[i]) ", terminator: " ")
```



```
}

print("\(arr.count)")

if arr.isEmpty {
    print("\n 陣列沒有元素")
} else {
    print("\n 陣列有元素")
}

arr.append(5.5)
for i in arr {
    print("\(i) ", terminator: " ")
}
print("")

arr.insert(6.6, at: 6)
for i in arr {
    print("\(i) ", terminator: " ")
}
print("")

arr.remove(at: 0)
for i in arr {
    print("\(i) ", terminator: " ")
}
print("")

arr.removeLast()
for i in arr {
    print("\(i) ", terminator: " ")
}
print("")

arr[2...4] = [66.6, 77.6, 88.6]
for i in arr {
    print("\(i) ", terminator: " ")
}
print("")

var arrInts = [Int]()
print("陣列中有 \(arrInts.count) 個")

arrInts.append(100)
//arrInts += [100]
print("陣列中有 \(arrInts.count) 個")
```

```
for i in arrInts {
    print(i)
}

arrInts = []
print("陣列中有 \(arrInts.count) 個")
var oneIntArray = [Double] (repeating: 1.23, count: 5)
for i in oneIntArray {
    print("\(i) ", terminator: " ")
}
print("")

var anotherIntArray = Array (repeating: 2.34, count: 5)
for i in anotherIntArray {
    print("\(i) ", terminator: " ")
}
print("")

var moreIntArray = oneIntArray + anotherIntArray
for i in moreIntArray {
    print("\(i) ", terminator: " ")
}
print("")
```

(d)

```
// dictionary
var nameScore = ["Jennifer": 92, "Amy": 90, "Linda":98]
var others = nameScore

print("在 nameScore 陣列: ")
for (name, score) in nameScore {
    print("\(name) : \(score)")
}
print("")

print("在 others 陣列: ")
for (name, score) in others {
    print("\(name) : \(score)")
}
print("")
```


(e)

```
var numbers = [1, 2, 3, 4, 5]
print(numbers[1...])
print(numbers[...2])
print(numbers[..<2])
print(numbers[...])
print(numbers[...])
numbers.swapAt(1, 4)
print(numbers)

for (index, value) in zip(1..., numbers) {
    print("Item \(index): \(value)")
}
```

(f)

```
let university = ["輔大", "東海", "交大"]
let landmark2 = ["中美堂", "東海教堂", "竹湖"]
let university_Landmark = zip(university, landmark2)
for data in university_Landmark {
    print(data)
}

let dic2 = Dictionary(uniqueKeysWithValues: university_Landmark)

for (university, landmark) in dic2 {
    print("University: \(university), Landmark: \(landmark)")
}
```

2. 以下的程式皆有些許的 bugs，請你 debug 一下，順便增加你程式設計的能力。

(a)

```
var countries = Dictionary<String: String>()
countries["France"] = "Eiffel Tower"
countries["Taiwan"] = "Taipei 101"
countries["Germany"] = "Berlin"
for (country, landmark) in countries {
    print("\(country): \(landmark)")
}
```

```
if let oldValue = countries.updatevalue("Berlin Wall", forKey: "Germany") {
    print("The old value for Germany was \(oldValue)")
}

for (country, landmark) in countries {
    print("\(country): \(landmark)")
}

countries[USA] = "Statue of Liberty"
countries["Germany"] = Nil
for (country, landmark) in countries {
    print("\(country): \(landmark)")
}


if let removeLandmark = countries.removeValueForKey("USA") {
    print("The remove landmark name is \(removeLandmark)")
} else {
    print("The dictionary does not contain a value for USA")
}

// empty dictionary
countries = []
countries["Taiwan"] = "Taipei 101"
for (country, landmark) in countries {
    print("\(country): \(landmark)")
}
```

(b)

```
var fruits = ["Apple", "Orange", "Banana"]
for (index, food) in fruits {
    print("Item \(index+1): \(food)")
}

let arrays = [100, 23, 44]
let newArray = arrays.sort
for data in newArray {
    print("\(data) ", terminator: " ")
}
print("")
```


7 CHAPTER 函式

函式 (function) 是執行某一特定任務的片段程式。函式的主要目的是將程式予以模組化，減少重複性，並達到分工合作，以及利於維護。因為軟體開發成本約有四分之三是用於維護成本，所以提高維護性 (maintainability) 是很重要的。

當程式有重複的程式碼，或要將程式加以模組化時，就可使用函式處理之。Swift 的函式較其它程式語言提供更多的功能，如函式可以回傳多個值、提供外部參數名稱以易於閱讀、inout 參數可以修改參數值，以及可將函式型態當做參數型態或是回傳型態。這些主題將在本章加以論述。現從人人所知曉的九九乘法表開始。

7.1 定義與呼叫函式

假設要輸出小時候每人必須背的九九乘法表，如下表所示：

```
************************************************
   1   2   3   4   5   6   7   8   9
   2   4   6   8  10  12  14  16  18
   3   6   9  12  15  18  21  24  27
   4   8  12  16  20  24  28  32  36
   5  10  15  20  25  30  35  40  45
   6  12  18  24  30  36  42  48  54
   7  14  21  28  35  42  49  56  63
   8  16  24  32  40  48  56  64  72
   9  18  27  36  45  54  63  72  81
************************************************
```