

Swift function


指令集形成函數，請在第3行以func宣告函數numSquare，函數主體使用print列印num與num的乘積，函數主體可以使用已經在第2行宣告的變數num。本題中，函數numSquare既沒有輸出參數也沒有輸入參數

```
1 import UIKit
2 var num = 3
3 func            {
4     print(num * num)
5 }
```

3

"9\n"

第3行到第5行已完成函數numSquare()的宣告與函數主體設計，請在第6行呼叫函數numSquare()

```
1 import UIKit
2 var num = 3
3 func numSquare(){
4     print(num * num)
5 }
6 
```

3

"9\n"

請在第3行以func宣告函數numSquare，指定輸入參數外部(external)名稱為num，型態為Int。本題中，函數numSquare有單一輸入參數，但沒有輸出參數

```
1 import UIKit
2 var a = 8
3 func ██████████ {
4     print(num * num)
5 }
```

8

"64\n"

第3行到第5行已完成函數numSquare()的宣告與函數主體設計，請在第6行呼叫函數numSquare()，呼叫時請將輸入參數名稱設為num與內容指定為a

```
1 import UIKit
2 var a = 8
3 func ██████████ {
4     print(num * num)
5 }
6 ██████████
```

本題輸入

8

"64\n"

請在第3行，完成以func宣告函數numSquare，請指定輸入參數的外部(external)名稱為num，內部名稱為x。本題中，函數numSquare有單一輸入參數，型態為Int，但沒有輸出參數

```
1 import UIKit
2 var a = 8
3 func ██████████ {
4     print(x * x)
5 }
6 numSquare(num: a)
```

8

"64\n"

請在第3行以func宣告函數numSquare，指定輸入參數外部(external)名稱為num，型態為Int，並以符號“_”強迫呼叫時，不指定輸入參數名稱。本題中，函數numSquare有單一輸入參數，但沒有輸出參數

```
1 import UIKit
2 var a = 8
3 func ██████████ {
4     print(num * num)
5 }
6 numSquare(a)
```

8

"64\n"

請在第3行以func宣告函數numSquare，指定輸入參數外部(external)名稱為num，型態為Int，同時指定輸出參數型態為Int，函數主體使用return，回傳num與num的乘積。本題中，函數numSquare的宣告，同時指定輸入參數與輸出參數

```
1 import UIKit
2 var a = 8
3 func [REDACTED] {
4     [REDACTED]
5 }
```

8

64

第3行到第5行已完成函數numSquare()的宣告與函數主體設計，請在第6行宣告常數square，並呼叫函數numSquare()，以函數回傳值設定常數square的初始值，呼叫時請指定輸入參數名稱為num與內容指定為a

```
1 import UIKit
2 var a = 8
3 func [redacted] {
4     [redacted]
5 }
6 let [blue] [redacted]
```

8
64
64

本題輸入

本題宣告函數sumUp時，使用符號“_”，強迫呼叫時不指定輸入參數名稱，並將輸入參數nums的型態設定為整數陣列，以Int...表示，在呼叫時可以將輸入參數內容指定為多個整數，形成整數陣列，同時指定輸出參數型態為Int。請在第4行的for指令中，以變數num列舉陣列nums的內容，在第5行中將num加入total，在第7行中，以return回傳total之內容

```
1 import UIKit
2 func sumUp(_ nums: Int...) -> Int{
3     var total = 0
4     for [REDACTED] {
5         [REDACTED]
6     }
7     [REDACTED]
8 }
```

0

(3 times)

6

請在第9行宣告常數sum，並呼叫函數sumUp，傳入參數內容1, 2, 3，以函數sumUp的回傳值設定sum的初始值

```
1 import UIKit
2 func sumUp(_ nums: Int...) -> Int{
3     var total = 0
4     for [REDACTED] {
5         [REDACTED]
6     }
7     [REDACTED]
8 }
9 let [REDACTED]
```

本題輸入

```
0
(3 times)
6
6
```

請在第4行宣告函數printSum，在第5行將print函數的輸入參數設定為numA與numB的加總

```
1 import UIKit
2 var numA = 3
3 var numB = 2
4 func [REDACTED] {
5     [REDACTED]
6 }
7 printSum()
```

3

2

"5\n"

swap函數有兩個輸入參數，分別是a與b，型態設定為inout Int，函數主體內，參數a與b的更新結果將回傳，影響呼叫指令中輸入參數之內容。請完成第4行的函數宣告，以及第6行，第7行的主體命令，交換參數a與b的內容

```
1 import UIKit
2 var numA = 3
3 var numB = 2
4 func swap [REDACTED] {
5     let temp = a
6     [REDACTED]
7     [REDACTED]
8 }
```

3

2

3

2

3

請在第9行呼叫swap函數，將輸入參數a的內容指定為&numA，輸入參數b的內容指定為&numB。函數swap將回傳交換內容，更新變數numA與numB

```
1 import UIKit
2 var numA = 3
3 var numB = 2
4 func swap {
5     let temp = a
6
7
8 }
9
10 print(" numA:"+String(numA)+" numB:"+String(numB))
```

3
2
3
2
3

本題輸入

numA:2 numB:3

請在第2行完成函數宣告，輸入參數名稱為score，型態為Int，輸出參數型態為optional Int。請在第6行，以return回傳nil。請在第9行宣告bonus為常數，呼叫calculateBonus設定bonus常數初始值，呼叫時請將參數score的內容設定為90

```
1 import UIKit
2 func calculateBonus [redacted] {
3     if score >= 90{
4         return 20
5     } else {
6         [redacted]
7     }
8 }
9 if let [redacted] {
10     print(bonus)
11 }
```

20

"20\n"

請在第6行完成factorial函數設計

```
1 import UIKit
2 func factorial(_ num: Int) -> Int{
3     if num <= 1{
4         return 1
5     } else {
6         return           
7     }
8 }
9 print(factorial(5))|
```

1

(4 times)

"120\n"