Python: Week 3

Review

> 複習一下上次的內容！

for 迴圈 (for loop)：對每一個 ...

```
for i in range(5):
    print(i)

nba_teams = ["Boston Celtics", "Brooklyn Nets", "Golden State Warriors", "Phoenix Suns"]
for team in nba_teams:
    print(team)

for i in range(2):
    for j in range(2):
        print("i, j =", i, j)
```

while 迴圈：當 ....

當條件滿足時，就執行區塊裡面的內容。下面的範例依序印出 1 到 5。

```
# Printing numbers from 1 to 5
i = 1
while i <= 5:
    print(i)
    i += 1
```

一般而言，我們會有一個變數去控制條件，確保條件有一天會不滿足而結束迴圈 。不然就會陷入無窮迴圈裡面無法自拔。

控制次數的變數，一般稱為計數器 (Counter)，會在迴圈裏面被累加。

list (清單) 用來存放一連串的資料。

常見的操作我們會想要存取([])、新增(append)、刪除(remove)。也可以透過 + 和 * 運算。

```
fruits = ["apple", "banana", "cherry", "date"]
print(fruits)

# Accessing the first element in the list
print(fruits[0]) # Output: apple
```

```
# Adding an element to the end of the list
fruits.append("elderberry")
print(fruits) # Output: ["apple", "banana", "cherry", "date", "elderberry"]

fruits.remove("banana")
print(fruits) # Output: ["apple", "cherry", "date", "elderberry"]
```

區塊

縮排 (indentation) 很重要，這決定了程式的流程受不受條件或迴圈的影響。

```
if True
    # 被 if 約束
    # 被 if 約束
    # 被 if 約束

# 不被 if 管理
# 不被 if 管理
# 不被 if 管理


while True:
    # 重複做
    # 重複做
    # 重複做

# 做一次
# 做一次
# 做一次
```

進度條！！

印出金字塔的練習背後有一個有趣的延伸應用。

```
import time

total = 100
length = 50

for i in range(total + 1):
    percent = 100 * (i / float(total))
    filled_length = int(length * i // total)
    bar = "" * filled_length + "-" * (length - filled_length)
    print(f"\r|{bar}| {percent:.2f}% Complete", end="")
    time.sleep(0.1)  # Simulating work being done

print("\nDone!")
```

Function

什麼是函式(Function)？

函式是一段程式碼，可以被調用來執行特定的任務。使用函式的主要優點包括：

- 重用性: 函式可以在程式中多次調用，避免重複程式碼。
- 可讀性: 函式使程式碼更具結構性和可讀性。
- 模塊化: 函式將程式碼分解成更小、更可管理的部分。

定義函式

在 Python 中，函式使用 def 關鍵字來定義。下面是一個簡單的函式範例：

```
def greet(name):
    """
    這個函式用來問候一個人
    參數:
        name (str): 要問候的人的名字
    """
    print(f"Hello, {name}!")
```

調用函式

定義函式之後，可以通過函式名稱和參數來調用它：

```
greet("Alice")
```

帶有返回值的函式

函式可以返回一個值，使用 return 關鍵字：

```
def add(a, b):
    """
    這個函式返回兩個數的和
    參數:
        a (int/float): 第一個數
        b (int/float): 第二個數
    返回:
        int/float: 兩個數的和
    """
    return a + b
```

```
result = add(5, 3)
print(result)
```

帶有預設參數值的函式

函式參數可以有預設值，如果調用時未提供相應的參數，則使用預設值：

```
def greet(name, message="Hello"):
    """
    這個函式用來問候一個人，使用預設或自定義的問候語
    參數:
        name (str): 要問候的人的名字
        message (str): 問候語，預設為 "Hello"
    """
    print(f"{message}, {name}!")
```

```
greet("Bob")
greet("Bob", "Good morning")
```

進度條，但是 function 版

```
import time

def print_progress_bar(iteration, total, length=50):
    percent = 100 * (iteration / float(total))
    filled_length = int(length * iteration // total)
    bar = "" * filled_length + "-" * (length - filled_length)
    print(f"\r|{bar}| {percent:.2f}% Complete", end="")

# Example usage
total = 100
for i in range(total + 1):
    print_progress_bar(i, total)
    time.sleep(0.1)  # Simulating work being done

print("\nDone!")
```