Python Dictionary\_Set

Dictionary(字典)，同樣是一個容器(集合)可以用來存放不同資料型態的資料，不過與串列(List)、元組(Tuples)不一樣的地方是，它的每一個元素是以鍵(Key)及值(Value)構成，再由 {} 符號將所有元素括起來，如下範例：

* Dictionary(字典)有幾個特性：

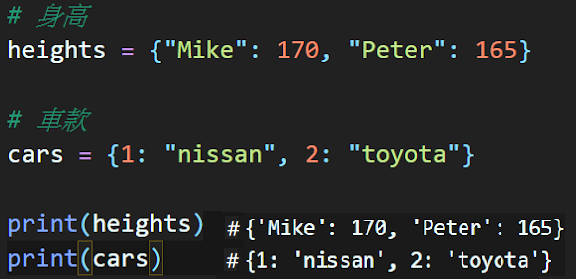
1. Iterable(可疊代的)：和前面介紹的[**字串(String)**](https://learncodewithmike.blogspot.com/2019/11/python4-python.html)、[**串列(List)**](https://learncodewithmike.blogspot.com/2019/12/python-list.html)及[**元組(Tuples)**](https://learncodewithmike.blogspot.com/2019/12/python-tuples.html)一樣是可疊代的物件，可以透過[**Python迴圈**](https://learncodewithmike.blogspot.com/2019/12/python.html)來進行元素的讀取。
2. Modifiable(可修改的)：和[**串列(List)**](https://learncodewithmike.blogspot.com/2019/12/python-list.html)一樣可以透過Python提供的方法(Method)來對Dictionary(字典)的值進行修改。
3. Key-Value pairs(鍵與值)：Dictionary(字典)的每一個元素由鍵(Key)及值(Value)構成。鍵(Key)的資料型態通常我們使用String(字串)或Integer(整數)，而值(Value)可以是任何資料型態。

瞭解了Dictionary(字典)的特性後，本篇來介紹Python Dictionary(字典)的基本操作，包含：

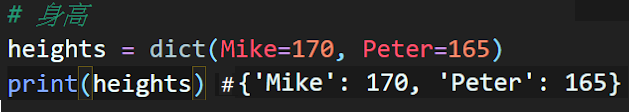
* 建立Dictionary的方法
* 存取Dictionary元素的方法
* 新增Dictionary元素的方法
* 修改Dictionary元素的方法
* 刪除Dictionary元素的方法
* 尋找Dictionary元素的方法

**一、建立Dictionary的方法**

1.於 {} 符號中輸入每個元素的鍵(Key)與值(Value)。

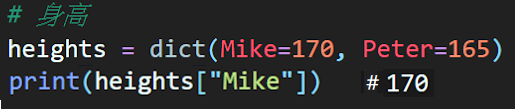


2.使用dict()方法，傳入鍵(Key)的名稱，並且指派值(Value)給它。此種傳入參數稱為關鍵字參數(keyword arguments)，後續會出文章詳加介紹。

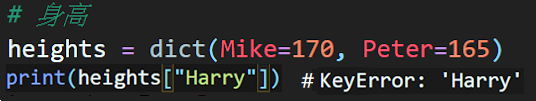


**二、存取Dictionary元素的方法**

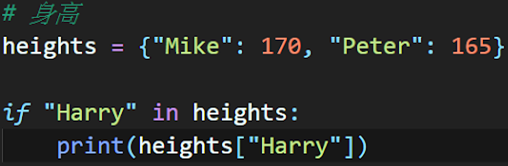
1.使用 [] 符號，傳入鍵(Key)的名稱。不像[**字串(String)**](https://learncodewithmike.blogspot.com/2019/11/python4-python.html)、[**串列(List)**](https://learncodewithmike.blogspot.com/2019/12/python-list.html)及[**元組(Tuples)**](https://learncodewithmike.blogspot.com/2019/12/python-tuples.html)是傳入位置索引值。



當存取的鍵(Key)名稱不在字典(Dictionary)中時，會發生KeyError的例外錯誤。

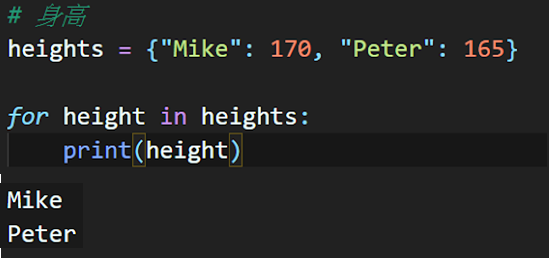


要解決此問題的方法有兩種，一種就是在存取字典(Dictionary)的元素前，先使用Python條件判斷來檢查元素是否在字典(Dictionary)中，如下範例。另一種解決方法就是使用文章最後會介紹的get()方法。

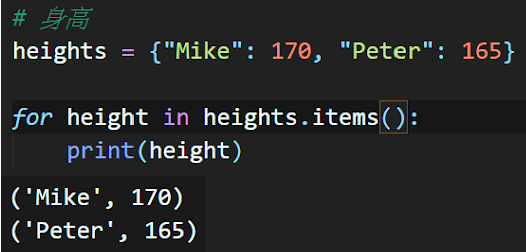


範例中由於Harry鍵(Key)名稱不存在於字典(Dictionary)中，所以不會印出它的值(Value)。

2.透過[**Python迴圈**](https://learncodewithmike.blogspot.com/2019/12/python.html)來存取字典(Dictionary)中的每一個元素。



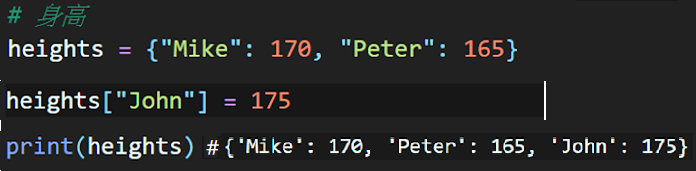
範例中可以看到，Python迴圈每一次讀取字典(Dictionary)時，只能存取到鍵(Key)的名稱，如果想要同時存取鍵(Key)與值(Value)的話，有兩種方法，第一種可以使用items()方法，如下範例，第二種方法則可以使用Python的Unpacking技巧(在下一篇文章中會來跟各位介紹)。



從執行結果可以看到，items()方法在迴圈每一次讀取時，會回傳一個[**Tuples(元組)**](https://learncodewithmike.blogspot.com/2019/12/python-tuples.html)，包含了鍵(Key)與值(Value)。

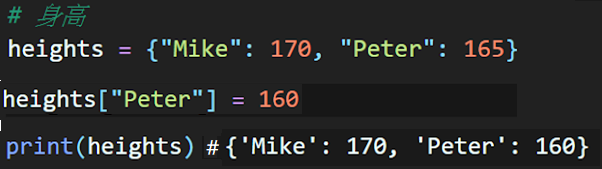
**三、新增Dictionary元素的方法**

1.於 [] 符號中輸入要新增的鍵(Key)的名稱，並且指派一個值(Value)給它。



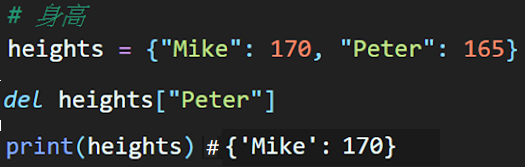
**四、修改Dictionary元素的方法**

1.於 [] 符號中輸入鍵(Key)的名稱，並且指派要修改的值(Value)給它。

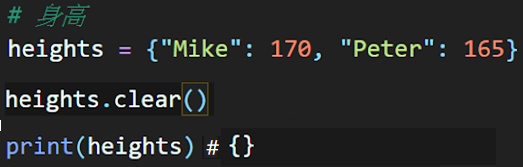


**五、刪除Dictionary元素的方法**

1.使用del指令，並且於 [] 符號中輸入要刪除的元素鍵(Key)名稱。



2.使用clear()方法，刪除字典(Dictionary)中的所有元素。

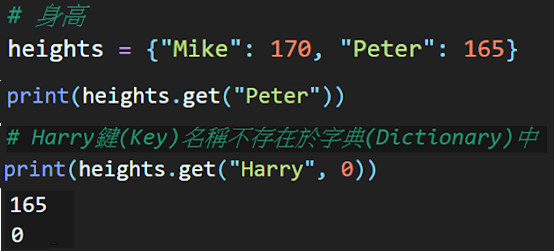


**六、尋找Dictionary元素的方法**

1.使用get()方法，傳入要尋找的鍵(Key)名稱，它會回傳其值(Value)。如果傳入的鍵(Key)名稱不存在，get()方法預設會回傳None。



get()方法也提供了第二個參數，來自行設定當鍵(Key)名稱不存在時，要回傳的值(Value)。



From：https://www.learncodewithmike.com/2019/12/python-dictionary.html

Dictionary 方法

|  |  |  |
| --- | --- | --- |
| **方法** | **說明** | **範例** |
| len(dictionary) | 返回字典中鍵值對的數量。 | len(my\_dict) |
| dictionary[key] | 通過鍵取得對應的值。 | value = my\_dict['key'] |
| dictionary[key] = value | 通過鍵設定新的鍵值對。 | my\_dict['new\_key'] ='new\_value' |
| key in dictionary | 檢查指定的鍵是否存在於字典中，返回布林值。 | if 'key' in my\_dict: |
| dictionary.get(key) | 返回指定鍵對應的值，如果鍵不存在，則返回指定的預設值。 | value = my\_dict.get('key', default) |
| dictionary.pop(key) | 刪除指定鍵的鍵值對，返回對應的值。 | removed\_value = my\_dict.pop('key') |
| dictionary.keys() | 返回字典中所有的鍵的列表。 | keys\_list = my\_dict.keys() |
| dictionary.values() | 返回字典中所有的值的列表。 | values\_list = my\_dict.values() |
| dictionary.items() | 返回字典中所有的鍵值對的元組列表。 | items\_list = my\_dict.items() |
| dictionary.clear() | 清空字典中的所有鍵值對，使其變為空字典。 | my\_dict.clear() |
| dictionary.copy() | 返回字典的副本。 | new\_dict = my\_dict.copy() |
| dictionary.update(other\_dict) | 將另一個字典的鍵值對更新到原字典中。 | my\_dict.update(new\_dict) |
| dictionary.setdefault(key, default) | 返回指定鍵對應的值，如果鍵不存在，則設定該鍵的值為預設值。 | value = my\_dict.setdefault('key', default) |

集合 (Set)

集合（Set）是 Python 中的一種無序、可變的資料結構，用於存儲多個元素，且集合中的元素是唯一的（不重複）。集合使用大括號 {} 來定義，元素之間使用逗號 , 分隔。

my\_set = {1, 2, 3}

another\_set = set([3, 4, 5])

集合有以下特點：

1. **元素無序**：集合中的元素沒有固定的順序，因此無法使用索引來訪問。
2. **元素唯一**：集合中的元素是唯一的，不會有重複的元素。
3. **可變性**：集合中的元素可以被新增、刪除和修改。

你可以使用集合來解決需要存儲一組元素，但不需要考慮元素順序和重複的問題的情況。  
集合提供了各種方法來執行集合間的交集、聯集、差集等操作。

**新增元素到集合**

my\_set = {1, 2, 3}

my\_set.add(4)

print(my\_set) # 輸出：{1, 2, 3, 4}

**刪除元素從集合**

my\_set = {1, 2, 3, 4}

my\_set.remove(3) # 元素3不存在於集合，會引發錯誤

print(my\_set) # 輸出：{1, 2, 4}

刪除元素但不引發錯誤，如果元素不存在：

my\_set = {1, 2, 3, 4}

my\_set.discard(5) # 元素5不存在於集合，不會引發錯誤

print(my\_set) # 輸出：{1, 2, 3, 4}

**清空集合**

my\_set = {1, 2, 3, 4}

my\_set.clear()

print(my\_set) # 輸出：set()

**複製集合**

original\_set = {1, 2, 3}

new\_set = original\_set.copy()

print(new\_set) # 輸出：{1, 2, 3}

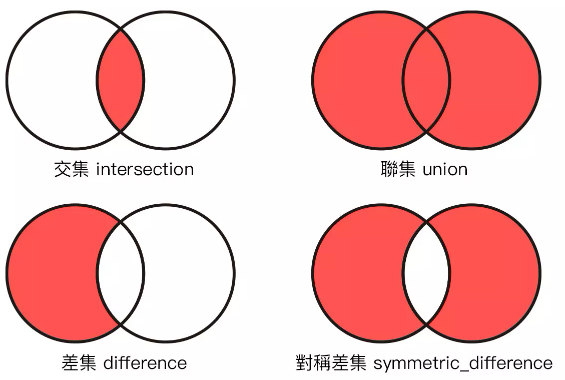
您可以根據這些範例來理解集合的新增、修改和刪除操作。

**集合運算**

| **運算** | **符號 / 方法** | **範例** |
| --- | --- | --- |
| AND交集 | a&b | set1 & set2 |
| 交集方法 | .intersection(set2) | set1.intersection(set2) |
| OR聯集 | a|b | set1 | set2 |
| 聯集方法 | .union(set2) | set1.union(set2) |
| XOR差集 | a-b | set1 - set2 |
| 差集方法 | .difference(set2) | set1.difference(set2) |
| 對稱差集 | a^b | set1 ^ set2 |
| 對稱差集方法 | .symmetric\_difference(set2) | a.symmetric\_difference(b) |

**交集、聯集、差集、對稱差集**

集合有四種運算型態，分別是「交集、聯集、差集、對稱差集」，透過下圖可以了解四種運算型態。

****

A&B

A-B

A|B

A^B

**集合常用方法**

| **方法 / 操作** | **說明** | **範例** |
| --- | --- | --- |
| add(element) | 新增元素到集合 | my\_set = {1, 2, 3} my\_set.add(4) # 現在 my\_set = {1, 2, 3, 4} |
| remove(element) | 移除元素從集合，如果元素不存在會引發錯誤 | my\_set = {1, 2, 3, 4} my\_set.remove(3) # 現在 my\_set = {1, 2, 4} |
| discard(element) | 移除元素從集合，元素不存在不會引發錯誤 | my\_set = {1, 2, 3, 4}  my\_set.discard(5)  # 現在 my\_set = {1, 2, 3, 4} |
| pop() | 移除並回傳任意元素 | my\_set = {1, 2, 3}  removed\_element = my\_set.pop() |
| clear() | 清空集合 | my\_set = {1, 2, 3}  my\_set.clear() # 現在 my\_set = set() |
| copy() | 複製集合 | original\_set = {1, 2, 3} new\_set = original\_set.copy() |
| len(set) | 回傳集合中元素的數量 | my\_set = {1, 2, 3}  length = len(my\_set) # length 為 3 |
| element in set | 檢查元素是否在集合中 | my\_set = {1, 2, 3}  result = 2 in my\_set # result 為 True |
| set1.union(set2) | 回傳兩個集合的聯集 | set1 = {1, 2, 3}  set2 = {3, 4, 5}  union\_set = set1.union(set2) |
| set1.intersection(set2) | 回傳兩個集合的交集 | set1 = {1, 2, 3} set2 = {3, 4, 5}  intersection\_set = set1.intersection(set2) |
| set1.difference(set2) | 回傳 set1 減去 set2 的差集 | set1 = {1, 2, 3}  set2 = {3, 4, 5}  difference\_set = set1.difference(set2) |
| set1.symmetric\_ difference(set2) | 回傳兩個集合的對稱差集 | set1 = {1, 2, 3} set2 = {3, 4, 5} symmetric\_difference\_set = set1.symmetric\_difference(set2) |
| set1.issubset(set2) | 檢查 set1 是否是 set2 的子集 | set1 = {1, 2} set2 = {1, 2, 3} result = set1.issubset(set2) # result 為 True |
| set1.issuperset(set2) | 檢查 set1 是否是 set2 的超集 | set1 = {1, 2, 3} set2 = {1, 2} result = set1.issuperset(set2) # result 為 True |
| max() | 最大值 |  |
| min() | 最小值 |  |
| sorted() | 傳回排序新  (原集合不變) |  |
| sum() | 總和 |  |
| enumerated() | 傳回連續整數配對的enumerated 物件 |  |
| update() | A.update(B)將集合B的元素加入集合A中。 | A={"Python", "java", "Kotlin"}  B={"Pytgon", "C"}  A.update(B)  print(A) #{'C', 'Kotlin', 'Python', 'Pytgon', 'java'}  print(B) #{'Pytgon', 'C'} |
| frozenset() |  |  |

set是可變的，frozenset是不可變的：

仍然可以執行：&, |, -, ^, ==, !=, in, not in

也可以使用 ：len(), min(), max()、sum(), copy()

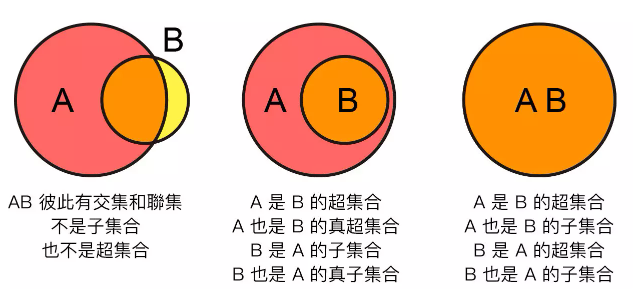
但不可以執行：add(), clear(), discard(), pop(), remove(), update()

<https://utrustcorp.com/python_dictionary_set/>

**子集合、超集合**

假設有 A、B 兩個集合，超集合和子集合的關係可以參考下圖：

| **集合** | **說明** |
| --- | --- |
| 超集合 | A 完全包含 B，A 和 B 所包含的元素可能完全相同 |
| 真超集合 | A 完全包含 B，且具有 B 沒有的的元素 |
| 子集合 | B 完全被 A 包含，A 和 B 所包含的元素可能完全相同 |
| 真子集合 | B 完全被 A 包含，且 A 具有 B 沒有的的元素 |



a = {1,2,3,4,5,6,7}

b = {3,4,5,6,7}

c = {1,2,3,4,5,6,7}

d = {6,7,8,9}

print(a<=a) # True 自己是自己的子集合

print(b<=a) # True b 是 a 的子集合

print(b<a) # True b 也是 a 的真子集合 ( 因爲沒有等於，完全包含 )

print(c<=a) # True c 是 a 的子集合

print(a<=c) # True a 也是 c 的子集合

print(d<a) # False d 和 a 沒有子集合或超集合關係

https://steam.oxxostudio.tw/category/python/basic/set.html