1.5. Assignment

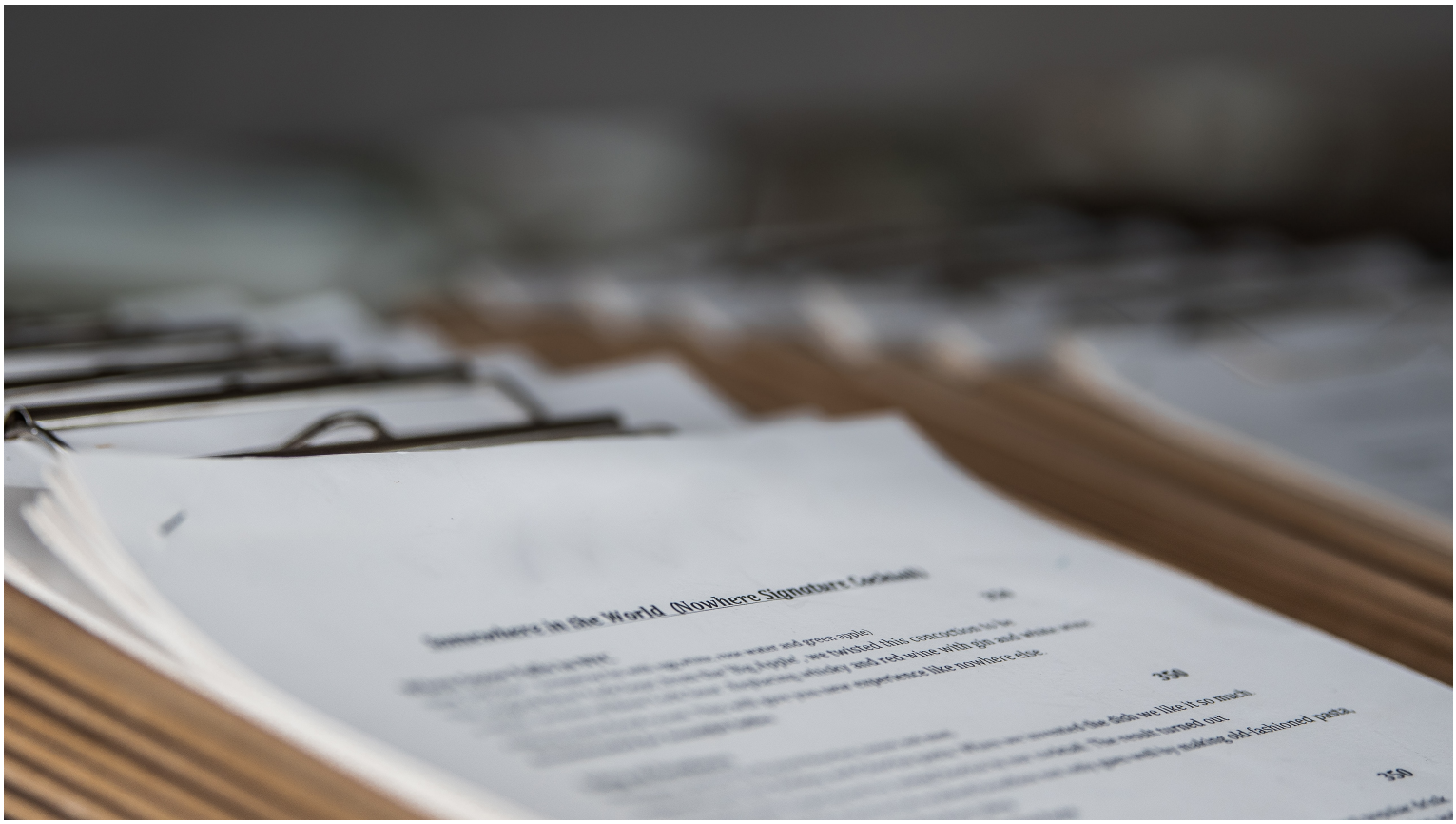


Fig. 1.5.1 Fig. 1.5.1 Photo by Arisa Chattasa on Unsplash

Note

Outline

1. Introduction
2. Examples
    a. Example 1: Integer
    b. Example 2: More Data Types

Note

Roadmap

1. This topic: Assignment

> myMaze
>
> | Basic Syntax | | | | | | | | | | | | | | | |
> |---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
> | Encoding | Data Type | | | | | | | Output | | | | | Input | | Remark | Identation |
> | | Number | | | | String | Container* | Type Casting | print() | Format | | | Escape | Assignment | input() | | |
> | | int | float | bool | complex* | | | | | % | format() | F-String | | | | | |

2. Course: Python 1
3. Subject: Programming
4. Field
    a. Software Engineering (SE)
    b. Computer Science and Information Engineering (CSIE)
    c. Electrical/Electronics Engineering (EE)

1.5.1. Introduction

1. 在許多程式語言中，都會看到「賦值」這個字眼，也就是將特定資料值指派給變數。
2. 注意：程式語言裡面的「=」不是數學邏輯，而是賦值的意思！

```
1  x = 12
```

3. 所以，上面這個運算式的意思就是將12 這個數值賦予給x 這個變數。
4. 從這行運算式之後，所有x的內容都裝著12，直到x 被賦予其他值為止。
5. 但是這裡需要注意一件事：Python 的變數又可分成全域變數(Global Variable)和區域變數(Local Variable)。
6. 全域變數 = 廣域變數
7. 顧名思義，全域變數就是整個Python 檔案都可以使用，變數只要賦值一次， 接下來所有使用到該變數名稱的程式碼便都會指向同一個物件， 上面所提到的情形就是屬於全域變數。
8. 反之，區域變數則大多是出現在迴圈與函數裡面3， 如果一個變數首次被賦值的地方是放在被迴圈或函數所【涵蓋】的區域裡， 那此變數的【效力範圍】就只存在該區域。
9. 一般而言，Python 屬於直譯式語言的一種。
10. 因此，在使用變數時並不需要事先宣告其型態；但是，當一個變數第一次出現時仍必須給予一個初始值。
11. 如同之前所說，變數所代表的是物件參照，未必一定要是數值型態的物件；因此，此處的賦值動作也未必是給予數值，可能是字串，或者集合型態的資料型態，如set、list 等等。
12. 如果有多個變數，也可以在一行程式碼中一次全賦值，如下所示：

1.5.2. Examples

> **Note**
>
> 寫程式哪有這麼難！掌握這8個重點，讓學習Coding效率加倍！
>
> 1. ALPHA Camp, 風傳媒, 20170326。 [Web]

1.5.2.1. Example 1: Integer

1. Code+Output

> **Code: Ex1** | Output
>
> Listing 1.5.2.1.1
>
> /src/Assignment/Ex1.py
>
> ```
> 1  iA = 10      # iA的變數被賦值為10
> 2  print(iA)    # 輸出變數的內容
> ```

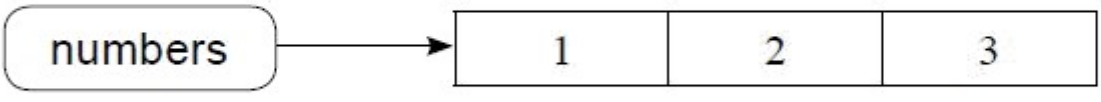
1.5.2.2. Example 2: More Data Types

1. Code+Output

> **Code** | Output
>
> Listing 1.5.2.2.1
>
> /src/Assignment/Ex2.py
>
> ```
> 1  integer1 = integer2 = integer3 = 10    # 多個變數可以同時賦予相同之值
> 2  string1, float1 = 'test', 12.0         # 多個不同型態的變數也可以分別賦予不同的值
> 3  print(integer1, integer2, integer3, string1, float1)
> ```

2. 從上面的例子可以看出，就算是不同型態的變數，也可以藉由「,」隔開，如此便可以在同一行程式碼中全部賦值。
3. 不過，基於閱讀方便，菲絲恩建議還是相同型態的變數才在同一行程式碼賦值，這樣在閱讀程式碼時才不容易產生混淆。

#. 當建立numbers 這個變數，並給予[1, 2, 3] 這個值，就好像建立出一個叫numbers 的框框，它會指向一組具有 [1, 2, 3] 值的 list。

> numbers = [1, 2, 3]
>
> 
>
>
> 圖 3-1 numbers 賦值時在記憶體中的可能排列方式

1. 如果再執行numbers=[4, 5, 6]，就會建立產生另一組值為[4, 5, 6] 的list，並且讓numbers 這個變數指向[4, 5, 6]。

2. 也就是說，Python 的賦值其實比較像是「指標」，每當要將值賦予給變數時，就是讓變數指向要賦予的值。

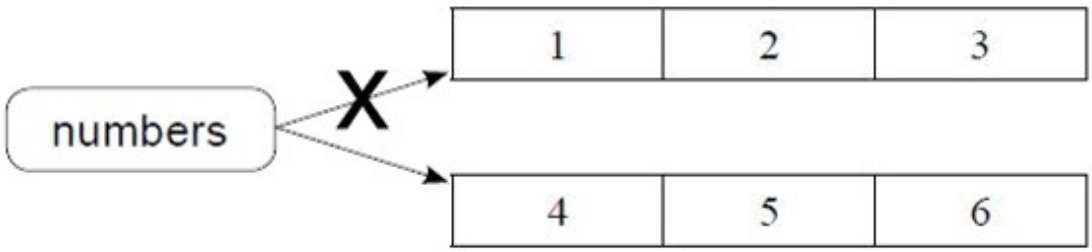


圖 3-2 numbers 再次賦值時在記憶體中的可能排列方式

3. 那麼，當執行完numbers = [4, 5, 6] 之後，剛剛[1, 2, 3] 的這個list 就沒有變數指向它。
4. 此時，這個list 就在垃圾回收機制的候選回收名單之中。
5. 記得前面所提到的，新增物件需要跟系統索取記憶體空間；但是記憶體有限，因此沒有被使用(指向) 到的物件，Python 就會自動將其回收，以清理出更多可用的記憶體空間。

Note

1. Start: 20120311
2. System Environment:

> Listing 1.5.2.2.2
>
> **requirements.txt**

```
sphinx>=6.1.3                           # Sphinx
graphviz>=0.20.1                        # Graphviz
sphinxbootstrap4theme>=0.6.0            # Theme: Bootstrap
sphinx-material>=0.0.35                 # Theme: Material
sphinxcontrib-plantuml>=0.25            # PlantUML
sphinxcontrib.bibtex>=2.5.0             # Bibliography
sphinx-autorun>=1.1.1                   # ExecCode: pycon
sphinx-execute-code-python3>=0.3        # ExecCode
btd.sphinx.inheritance-diagram>=2.3.1   # Diagram
sphinx-copybutton>=0.5.1                # Copy button
sphinx_code_tabs>=0.5.3                 # Tabs
sphinx-immaterial>=0.11.3               # Tabs

#-------------------------------------------------------
#-- Minor Extension
#-------------------------------------------------------
sphinxcontrib.httpdomain>=1.8.1         # HTTP API

#sphinxcontrib-blockdiag>=3.0.0         # Diagram: block
#sphinxcontrib-actdiag>=3.0.0           # Diagram: activity
#sphinxcontrib-nwdiag>=2.0.0            # Diagram: network
#sphinxcontrib-seqdiag>=3.0.0           # Diagram: sequence

#-------------------------------------------------------
#-- Still Wait For Upgrading Version
#-------------------------------------------------------

#-------------------------------------------------------
#-- Still Under Testing
#-------------------------------------------------------
#numpy>=1.24.2                          # Figure: numpy

#-------------------------------------------------------
#-- NOT Workable
#-------------------------------------------------------
#sphinxcontrib.jsdemo==0.1.4    # ExecCode: Need replace add_js_file()
#jupyter-sphinx==0.4.0          # ExecCode: Need gcc compiler
#sphinxcontrib.slide==1.0.0     # Slide: Slideshare
#hieroglyph==2.1.0              # Slide: make slides
#matplotlib>=3.7.1              # Plot: Need Python >= v3.8
#manim==0.17.2                  # Diagram: scipy, numpy need gcc
#sphinx_diagrams==0.4.0         # Diagram: Need GKE access
#sphinx-tabs>=3.4.1                     # Tabs: Conflict w/ sphinx-material
```