## Chapter 1. variables

※ Define :

Variable is a name that is used to refer to memory location. Python variable is also known as an identifier and used to hold value.

※features

a) It is recommended to use lowercase letters for the variable name.

b) The first character of the variable must be an alphabet or underscore ( \_ ).

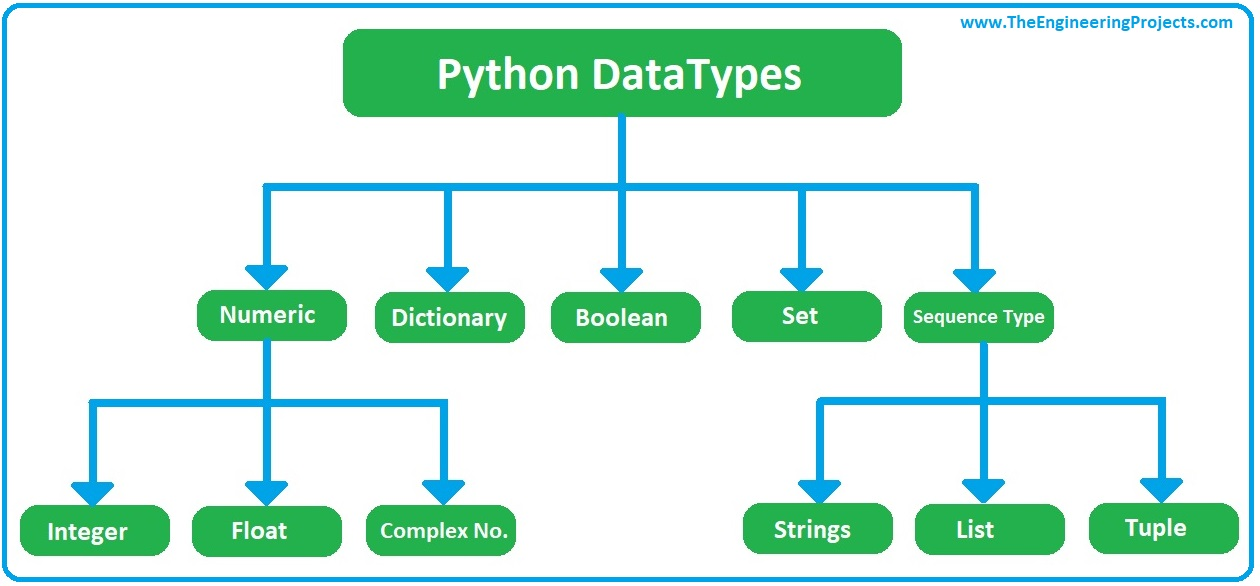
c) Identifier name must not contain any white-space, or special character (!, @, #, %, ^, &, \*).

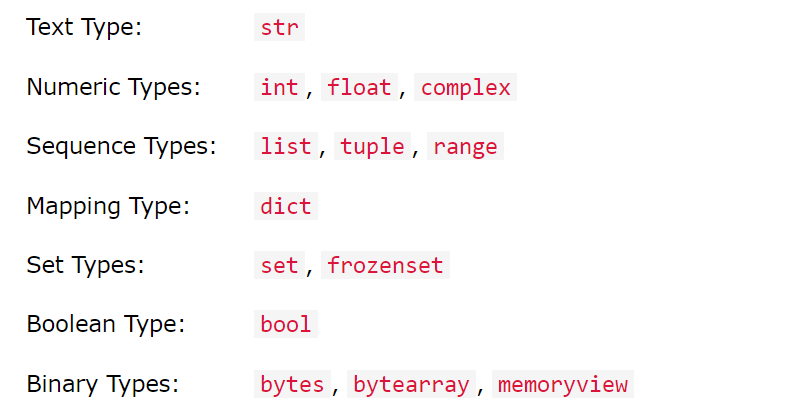
※Multiple Assignment

x=y=z=50

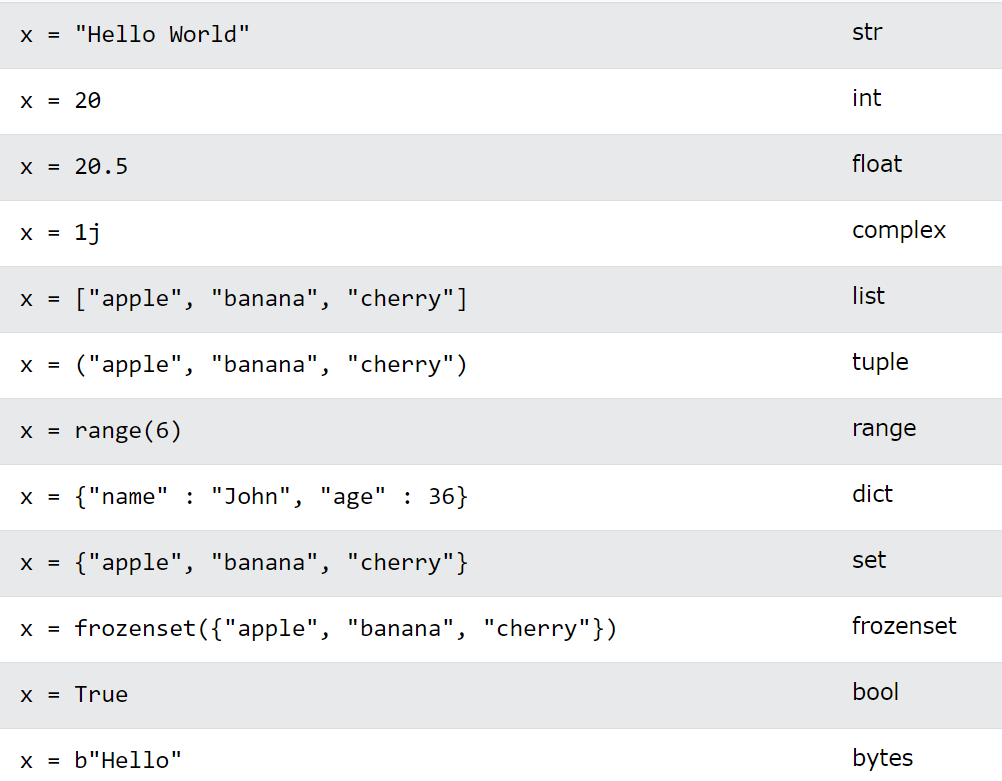
a,b,c=5,10,15

## chapter 2.data type





### 2.1 type example



### 2.2 print

*print(value(s), sep= ‘ ‘, end = ‘\n’, file=file, flush=flush)*

* Example 1

|  |
| --- |
| a = 10; b = 20  print(a,b)  print(type(a),type(b))  print (id(a,b)) #error  print(id(a),id(b)) |
| 10 20  <class 'int'> <class 'int'>  Error: id() takes exactly one argument (2 given) |

* Example 2 basic syntax

|  |
| --- |
| Print(var1,var2,sep=, end=)  ~sep=separate with  ~end =end with |

* Examples

|  |
| --- |
| 1. print('hello', 'world', sep='\n') #start a new line  hello  world |
| 2. print('home', 'user', 'documents', sep='/')  home/user/documents |
| 3.repeat value  #when var is number, it is multi by  #when var is string,  a = 10; b = 20; c='abc'  Print(b\*3) #90  Print(c\*3) #abcabcabc |
| 4. print dictionary  Dictionary syntax  d = {key1 : value1, key2 : value2 }  separate with “,”  dit = {'a': 1, 'b': 2, 'b': '3'}  print(dit)  print(dit['b'])  #result:  {'a': 1, 'b': '3'}  3 |
|  |

### 2.4 print advance syntax

# : %s means string; %d means integer; %f means floating

#: %5d print 5 digits, if less 5 digits, then fill with blank char; if more than 5, print all

#: %5s: print 5 chars string, ………(above)

#: %8.2f: print 8 digits, in which 2 decimal points, 5 integer, 如果唔夠，fill with 0

|  |
| --- |
| 1. include a number and a string  name = "John"  age = 23  print("%s is %d years old." % (name, age)) |
| John is 23 years old. |

* Example 2 print a list

|  |
| --- |
| mylist = [1,2,3]  print("A list: %s" % mylist)  print ("A list:%s" % mylist[1:]) |
| A list: [1, 2, 3]  A list:[2, 3] |

* Print floating point

|  |
| --- |
| Question : Hello John Doe. Your current balance is $53.44.  **Give:** **data = ("John", "Doe", 53.44)**  **format\_string = "Hello"** |
| data = ("John", "Doe", 53.44)  format\_string = "Hello %s %s. Your current balance is $%s."  print(format\_string % data)  #result: Hello John Doe. Your current balance is $53.44. |

### 2.3 input

By default input() function takes the user’s input in a string. So, to take the input in the form of int you need to use int() along with the input function.

* Examples

|  |
| --- |
| 1.basic input  # Taking input from the user  name = input("Enter your name: ")    # Output  print("Hello", name)  #result:  Enter your name: kitty  Hello kitty |
| 2.error , 小心  score=input("input num a : ")  print(score+10)  #result:  input num a : 20  **can only concatenate str (not "int") to str** |
| 3.input 食到int/ list 其他非 string type  score=int(input("input num a : "))  print(score+10)  result:  input num a : 20  30 |
|  |

### 2.4 write code in multi-lines

* Example 1

|  |
| --- |
| y = 3; x = 5; print(x+y) |
| y = 3  x = 5  print(x+y) |

* Example 2

Long string use triple quote’’’

|  |
| --- |
| s = '''Mistress! what, mistress! Juliet! fast, I warrant her, she:  Why, lamb! why, lady! fie, you slug-a-bed!  Why, love, I say! madam! sweet-heart! why, bride!  What, not a word? you take your pennyworths now;  Sleep for a week; for the next night, I warrant''' |
| Print (s) |

* a long variable use backslash\

|  |
| --- |
| . a = 1 + 2 + 3 + 4 - 5 \* 2  b = 1 \  + 2 + \  3 + 4\  - 5 \*\  2  print(a == b) |
| math\_result = 1 + 2 + 3 + 4 + \  5 + 6 + 7 + 8 + \  9 + 10  print(math\_result) |

* break to many line

|  |
| --- |
| message = ("Hello\n"  "Hi\n"  "Namaste")  print(message) |
| Hello  Hi  Namaste |

* keep in a SAME line

|  |
| --- |
| m="hello\  kitty"  print(m) |
| hello kitty |

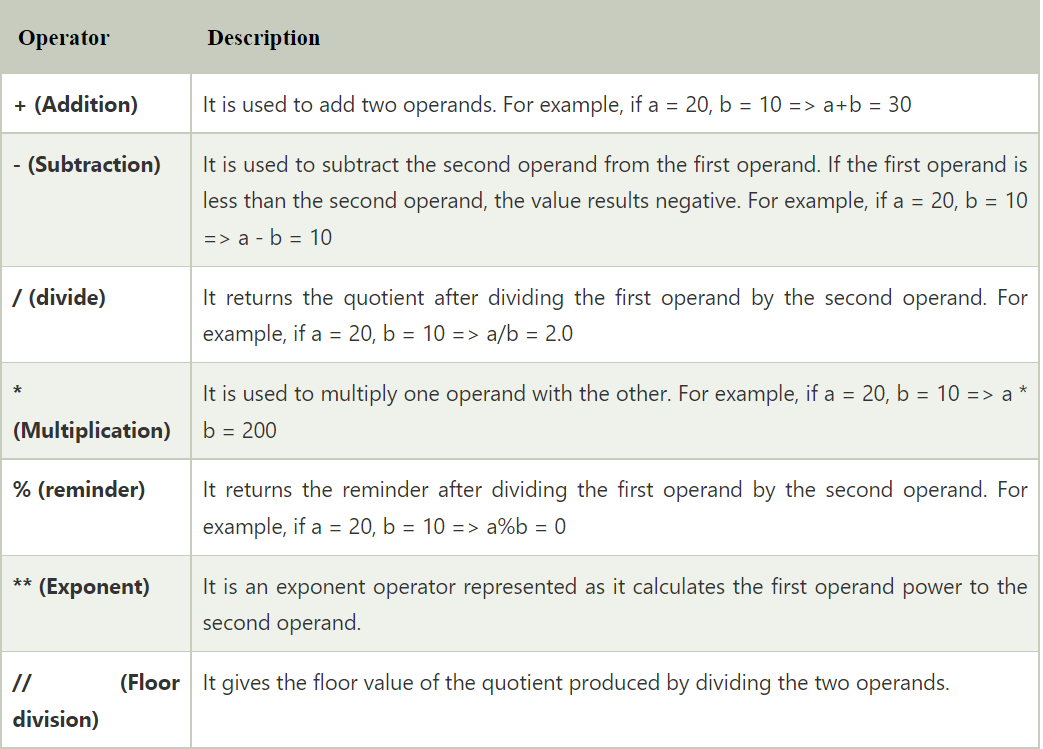
# Chapter 3.operator

## 3.1 background

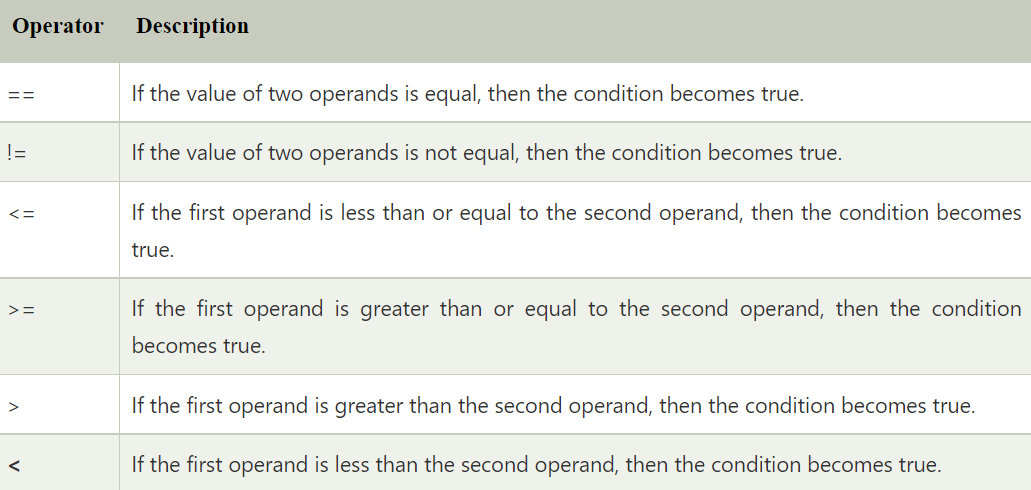
The operator can be defined as a symbol which is responsible for a particular operation between two operands.

* Arithmetic operators
* Comparison operators
* Assignment Operators
* Logical Operators
* Bitwise Operators
* Membership Operators
* Identity Operators

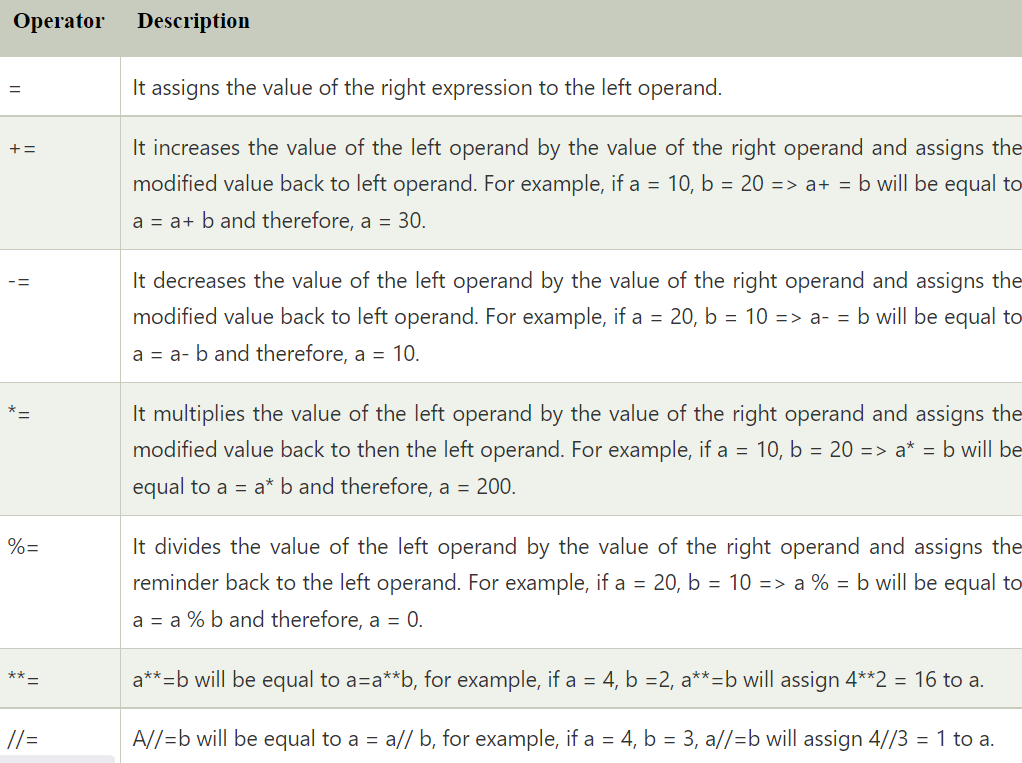
## 3.2 Arithmetic Operators



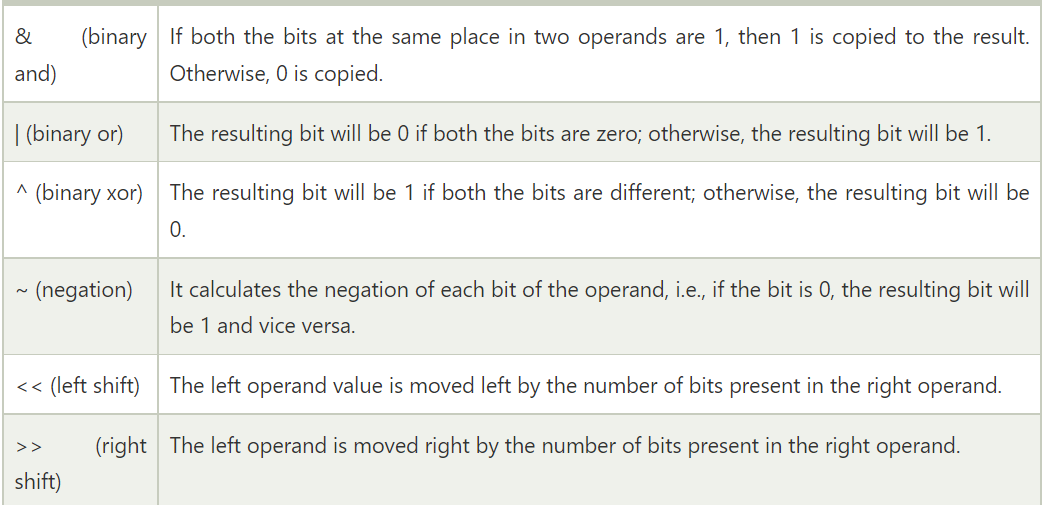
## 3.2 Comparison operator



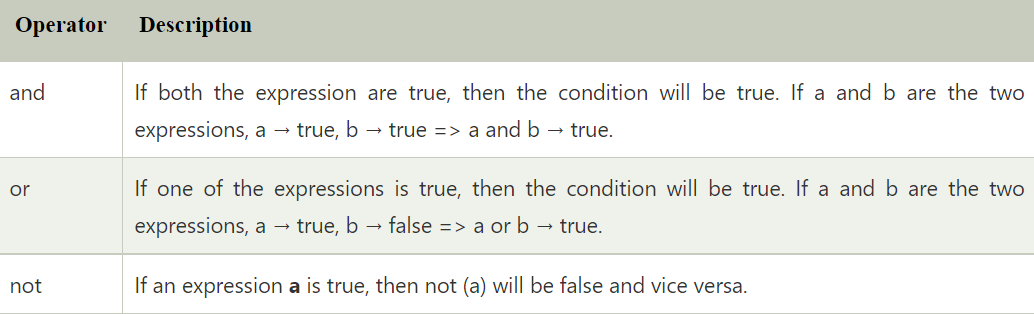
## 3.3 Assignment Operators



## 3.4 Bitwise Operators



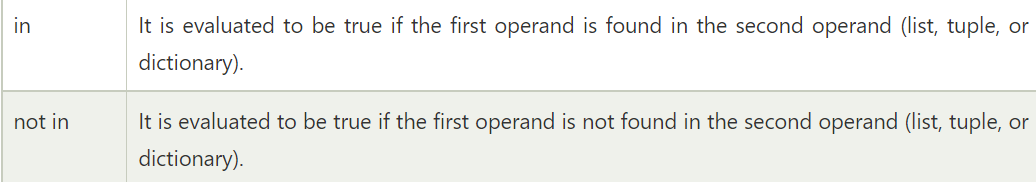
## 3.5 Logical Operators



## 3.6 Membership Operators

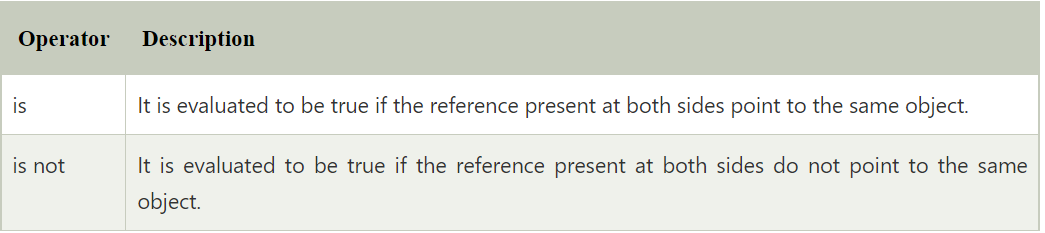
Python membership operators are used to check the membership of value inside a Python data structure.

If the value is present in the data structure, then the resulting value is true otherwise it returns false.



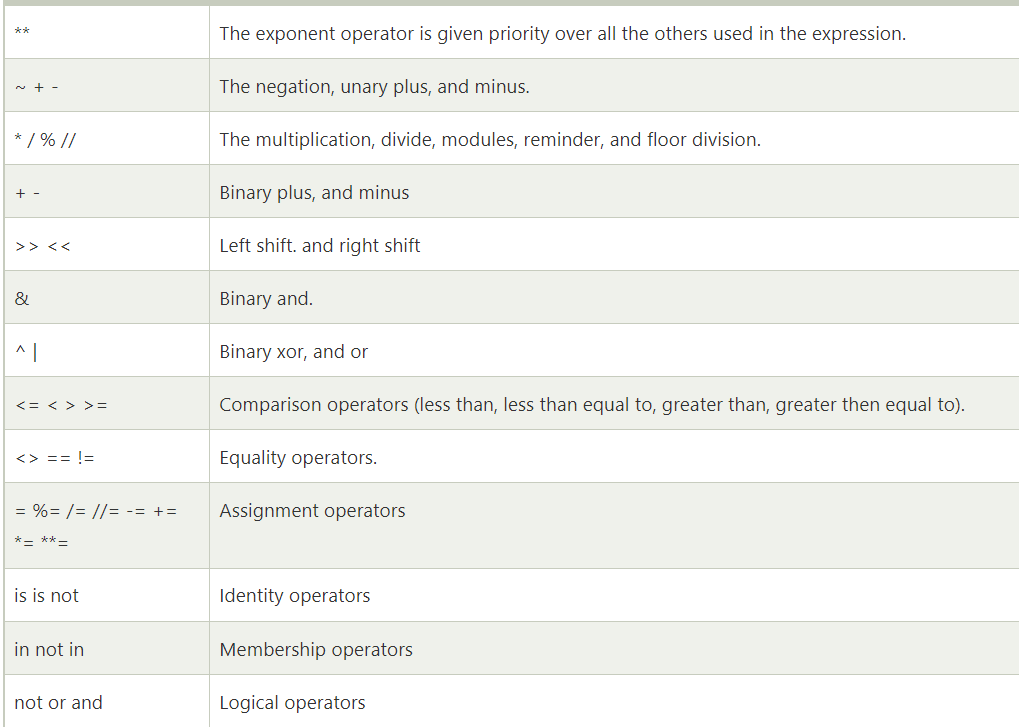
## 3.7 Identity Operators

The identity operators are used to decide whether an element certain class or type.



## 3.8 Operator Precedence

The precedence of the operators is essential to find out since it enables us to know which operator should be evaluated first.



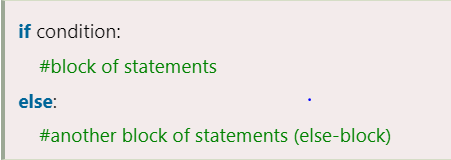
## 3.9 examples

### 3.9.1 arithmetic cases

|  |
| --- |
| 1. simple operator  i=6  i+=3 #i=i+3  print (i) #9  i-=1 #i=i-1  print (i) #i=8  i/=2  print(i) #4.0  #i="{:.2f}".format(i)  print ("{:.2f}".format(i)) #4.00  #2 decimal point (2 floating points) |
| 2. advance operators  my\_float = 2.13456  print(round(my\_float,3)) #2.135 |
| 3. calculation  i=32  print (i/5) #除法 6.4  print (i//5) #商 6  print(i%5) #餘數 2  print(i\*\*3) #32\*32\*32=32768  print(i\*3) #32\*3=96 |
|  |

## Chapter :4. if

### 4.1: syntax





|  |
| --- |
| marks = int(input("Enter the marks? "))  if marks > 85 and marks <= 100:  print("Congrats ! you scored grade A ...")  elif marks > 60 and marks <= 85:  print("You scored grade B + ...")  elif marks > 40 and marks <= 60:  print("You scored grade B ...")  elif (marks > 30 and marks <= 40):  print("You scored grade C ...")  else:  print("Sorry you are fail ?") |
| Enter the marks? 77  You scored grade B + ... |

### 4.2.question

商場：買夠10k 打8折，買滿5k 打85折，買滿3000打 9折，買滿1000打95折

|  |
| --- |
| budget = int(input("how munch do you buy : "))  if budget >= 10000:  print("need to pay: " ,budget\*0.8)  elif budget >= 5000:  print("need to pay: " , budget\*0.85)  elif budget >= 5000:  print("need to pay: " ,budget\*0.9)  elif budget >=1000:  print("need to pay: " , budget\*0.95)  else:  print("need to pay:" , budget) |
| #result:  how munch do you buy : 7200  need to pay: 6120.0 |

Remark:

|  |
| --- |
| 不可用  print("need to pay: " +budget\*0.85)  error: concat is only for str  除非轉成：change number to string  print("need to pay: " + str(budget\*0.8)) |

## Chapter 5 :list + Range

### 5.1 background

※Python lists are mutable type its mean we can modify its element after it created.

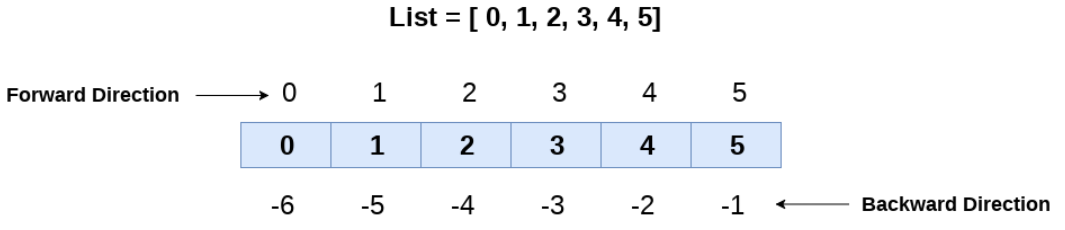
※A list can be defined as a collection of values or items of different types.

※The items in the list are separated with the comma (,) and enclosed with the square brackets [].

|  |
| --- |
| L1 = ["John", 102, "USA"]  L2 = [1, 2, 3, 4, 5, 6]  print(L1)  print(type(L1))  print(id(L1))  ##################################  ['John', 102, 'USA']  <class 'list'>  1469022132800 |

* Features
* The lists are ordered.
* The element of the list can access by index.
* The lists are the mutable type.
* The lists are mutable types.
* A list can store the number of various elements.

### 5.2 list extraction syntax



|  |
| --- |
| list = ['A',2,'c',4,'E']  print(list[-1])  print(list[-3:])  print(list[:-1])  print(list[-3:-1])  #####################################################  E  ['c', 4, 'E']  ['A', 2, 'c', 4]  ['c', 4] |

### 5.3 editing list

|  |
| --- |
| 1. list = [1, 2, 3, 4, 5, 6] 2. **print**(list) 3. # It will assign value to the value to second index 4. list[2] = 10 5. **print**(list) 6. # Adding multiple element 7. list[1:3] = [89, 78] 8. **print**(list) 9. # It will add value at the end of the list 10. list[-1] = 25 11. **print**(list) |
| [1, 2, 3, 4, 5, 6]  [1, 2, 10, 4, 5, 6]  [1, 89, 78, 4, 5, 6]  [1, 89, 78, 4, 5, 25] |

* **list index +min+max+ 加減**

**GIVEN : list1=[1,2,3,4,5,6]**

**list2=[‘e’,’f’,’g’,’h’]**

**x=[8,9]**

|  |
| --- |
| **1.** |
|  |
|  |
|  |
|  |

### 5.4 range

* 1.Syntax:

range(start, stop, step)

position to start. Default is 0

the incrementation. Default is 1

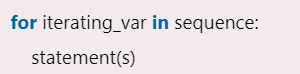
* 2.real-case

|  |
| --- |
| 1. r=range(6)  print(r)  #############  range(0, 6)  不是：0,1,2,4,5,6 |
| 2. r=range(3,10,2)  print(list(r))  TypeError: 'list' object is not callable  #so 要配合for loop |
| 3.雖然無法直接print, 但要清楚背後list logic  R=Rang(8,3,-1)  #list(r)=[8,7,6,5,4]  R=rang(3,8,2)  #list(R)=[3,5,7] |

## Chapter 6 for loop

### 6.1 background

#要識埋go, continue, break,pass



|  |
| --- |
| 1. str = "Python"  for i in str:  print (i)  ##################  P  y  t  h  o  n |
| 2.  str = "Python"  for i in str:  # print (i)  print(i, end=",") #sep=”, 不可以”  P,y,t,h,o,n, |

### 6.2 real-case

* Print a range

|  |
| --- |
| for x in range(2, 6):   print(x,end=” ”)  3 5 7 |

* 2. For +else :=final

|  |
| --- |
| for x in range(6):  print(x,end=",")  else:  print("Finally finished!")  0,1,2,3,4,5,Finally finished! |

* 3.Break

|  |
| --- |
| ##If the loop breaks, the else block is not executed.  for x in range(6):  if x == 3: break  print(x,end="-")  else:  print("Finally finished!")  0-1-2- |

* 4.2-D LOOP

|  |
| --- |
| #Both are string list，so can use + to concat  adj = ["red", "big", "tasty"]  fruits = ["apple", "banana", "cherry"]  for x in adj:  for y in fruits:  print(x+" +" +y)  ################################################  red +apple  red +banana  red +cherry  big +apple  big +banana  big +cherry  tasty +apple  tasty +banana  tasty +cherry |

* 5.Program to print the sum of the given list.

|  |
| --- |
| #print sum 就會return final value  list = [10,30,23,43,65,12]  sum = 0  for i in list:  sum = sum+i  print("The sum is:",sum)  The sum is: 183 |

### 6.3 interview questions

* 計算2-20，單數總和

|  |
| --- |
| sum=0  for i in range(2+1,20,2):  sum=sum+i  print(sum)  99 |

* 輸入一個整數，計算1到該整數的和

|  |
| --- |
| sum = 0  n = int(input("enter an integer: ")) #一定要=datatype(input())  for i in range(1, n+1, 1):  sum = sum+i  else :  print("total is :%d "%(sum))  enter an integer: 10  total is :55 |

* Program to number pyramid

Hints: use 2-d for-loop(nested loop)

|  |
| --- |
| 1  22  333  4444  55555  ………  nnnnnn |
| rows = int(input("Enter the rows: "))  for i in range(0,rows+1):  for j in range(i):  print(i,end = “ '')  print() # break line when every nested loop finished |
| # 半正確：  rows = int(input("Enter the rows: "))  for i in range(0,rows+1):  for j in range(i):  print(i,end=" ")  Enter the rows: 4  1 2 2 3 3 3 4 4 4 4 |

### 6.4 break & continue

* 1.define

1/～Break Statement - When the break statement is encountered, it brings control out of the loop.

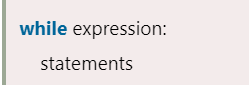
2/～Continue: exit this time loop and skip the rest of code of this this.

3/~ **Pass Statement -**The pass statement is used to declare the empty loop. It is also used to define empty class, function, and control statement.

When an external condition is triggered, the pass statement **allows you to handle the condition without the loop being impacted in any way;** all of the code will continue to be read unless a break or other statement occurs.

### 6.5 while loop

會有n=n+1, 因為for loop <> for loop,有increment.



while loop allows a part of the code to be executed until the given condition returns false. It is also known as a pre-tested loop.

* Example 1

# prints all letters except 'a' and 't' , s1=’javatpoint’

|  |
| --- |
| i = 0  str1 = 'javatpoint'  while i < len(str1):  if str1[i] == 'a' or str1[i] == 't':  i += 1  continue  print('Current Letter :', str1[i])  i += 1  ##############################  Current Letter : j  Current Letter : v  Current Letter : p  Current Letter : o  Current Letter : i  Current Letter : n |

* Example 2:

# when break statement soon it sees t , s1=’javatpoint’

|  |
| --- |
| i = 0  str1 = 'javatpoint'  while i < len(str1):  if str1[i] == 't':  i += 1  break  print('Current Letter :', str1[i])  i += 1  Current Letter : j  Current Letter : a  Current Letter : v  Current Letter : a |

### 6.6 Supplement examples:

|  |
| --- |
| number = 0  for number in range(10):  if number == 5:  break # break here  print('Number is ' + str(number))  print('Out of loop') |
| number = 0  for number in range(10):  if number == 5:  pass # pass here    print(str(number),end=',')  print('Out of loop')  0,1,2,3,4,5,6,7,8,9,Out of loop |
|  |

### 6.7 advance syntax of list & range

Given: L1=[1,2,3,4,5,6] L2=[‘b’,’c,’e’,’g’,’f’] x[8,9]

|  |  |  |
| --- | --- | --- |
| **List value extraction:**  **LIST VLAUE USE []** | | |
| L3=L1[2:4] | Range values in a list |  |
| L4=L1[1::2] | Range value with step |  |
| L5=max(L1)  L6=min(L1) | Max  Min value |  |
| L7=L1.index(3)  (x)L7=L1.index(3:5) | Index |  |
|  |  |  |
| **DML of list value**  **List Function use()** | | |
| L1[3]=100  Donot use index | update |  |
| L1.append(20) | Add a new value 在最後  (20) is value not index of list | 直接：L1.Aappend(20)  Not need:  L1=L1.append(20) |
| L1.insert(0,20) | 加插系指定位置value  Insert(pos, value) |  |
| L1.remove(1) | Delete a value |  |
|  |  |  |
| **Ordering value** | | |
| L1.sort() | 由小到大排列 |  |
| Reverse:  按照index倒序  L1=[2,3,4,5]  print(L1)  L1.reverse()  print(L1)  ###############  [2, 3, 4, 5]  [5, 4, 3, 2]  接入新list,要用reversed, past tense， ed   * 方法1：(不正確方法)   L1=[2,3,4,5]  L2=reversed(L1)  print(L2) # return id  print(list(L2)) # return value  <list\_reverseiterator object at 0x0000027933633940>  ###################################  print ctypes.cast(id(L2), ctypes.py\_object).value  [5, 4, 3, 2]   * 方法2：(reversed)   直接list(reverse(L1))  L1=[2,3,4,5]  L2=list(reversed(L1))  print(L2)  [5, 4, 3, 2]   * Smart 方法：L1[::-1]   digits = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]  # Rely on default offset values  d2=digits[::-1]  print(d2)  [9, 8, 7, 6, 5, 4, 3, 2, 1, 0]  **Remark: 仲可以抽 單雙數出來** | | |

## Chapter 7: 第一次總結：

### Q1: check 有冇contain a value,

而且要指出位置（index）

Status: not completed

|  |
| --- |
| L1 = ['A', 'B', 'C', 'D', 'E', 'F']  #no need L1=list[xxxx]  count = 1  n = str(input("alpha what to check:"))  for i in L1:  if i == n:  print("the list cotain")  count = count+1  print("the %s is locate in " % list[count]+str(count))  break  else :  print("can not find") |

### Q2: for each loop vs standard for loop

* Range(0,len(x),n):

Print(x[i])

|  |
| --- |
| t1 = (10, 20, 30, 40, 50, 60)  print(t1)  for i in t1:  print(i,end="/ ")  print()  for i in range(0,len(t1),2):  print(t1[i],end='/') #still similar with java  (10, 20, 30, 40, 50, 60)  10/ 20/ 30/ 40/ 50/ 60/  10/30/50/ |

* Q3: print list in reverse order

|  |
| --- |
| t2=('A','B','C','D','E',)  print(t2)  print()  ###順序  for i in range(0,len(t2),1):  print(t2[i],end=";")  print()  ####用reversed function  t3=list(reversed(t2))  print(t3)  print()  **####用for loop 倒序**  for i in range(len(t2)-1,0,-1):  print (t2[i],end=';')  ('A', 'B', 'C', 'D', 'E')  A;B;C;D;E;  ['E', 'D', 'C', 'B', 'A']  E;D;C;B; |

## Chapter 8: tuple

### 8.1 background:

1/:Tuple is used to store the sequence of immutable Python objects.

2/:tuple use (), list use []

3/:Note: The tuple which is created without using parentheses is also known as tuple packing.

* Example:

|  |
| --- |
| #DON’T know why T3 is tuple  T1 = (101, "Peter", 22)  T2 = ("Apple", "Banana", "Orange")  T3 = 10,20,30,40,50  print(T1);print(type(T1));print(type(T3))  (101, 'Peter', 22)  <class 'tuple'>  <class 'tuple'> |

### 8.2 tuple vs list

* Ex1:

|  |
| --- |
| list1 = ['JavaTpoint', 1, 2, 54.30, {'Name: ''Peter'}]  print(type(list))  tuple1 = ('JavaTpoint',5,8,31.9,[1,2,3])  print(type(tuple1))  <class 'type'>  <class 'tuple'> |

## Chapter 9: set (集合)

### 9.1 background

1.set is the collection of the unordered items. (每次print order 會變)

2.Each element in the set must be unique, immutable, and the sets remove the duplicate elements.

3.Sets are mutable which means we can modify it after its creation.(new memory ref)

4. The major advantage of using a set, as opposed to a list, is that it has a highly optimized method for checking whether a specific element is contained in the set.

### 9.2 set DML syntax

* Ex1:

|  |
| --- |
| Set can use {} or set([xxxxx])  如果：s2=set(2,"B","QQQ",'TVq'), no []  Error: set expected at most 1 argument, got 4  除非，s2=set(“ssss”) ，only has 1 elements |

* Ex2: create sets

1:curly brace{};

2:use set () method

|  |
| --- |
| s1={1,'A',7,"ban"}  print(s1)  print("s1 is a ", type(s1))  s2=set([2,"B","QQQ",'TVq'])  print(s2)  print("s2 is a ",type)  ##################  {1, 'ban', 7, 'A'}  s1 is a <class 'set'>  {'TVq', 2, 'QQQ', 'B'}  s2 is a <class 'type'> |

* Ex3:remove duplicate of element of List

|  |
| --- |
| L1=["a","a","2","2"]  print(type(L1))  s1=set(L1)  print("s1 is a ", type(s1))  print(s1)  ###########################  <class 'list'>  s1 is a <class 'set'>  {'a', '2'} |

* DML of set

|  |  |
| --- | --- |
| Add | s1=set([1,2,'CC',"dd","EGG"])  s1=s1.add("333")  print(s1)  #####  None  直接：s1.add(“333”)  Print(s1) |
| Remove  discard | Both are delete dedicated element, but remove returns an error if the element is not in the set. |
| Clear | s1=["a","a","2","2"]  s2=s1.clear()  print(s2)  ###########  None |
| Update | We cannot access or change an element of a set using indexing or slicing. So update is an alternatives of copying。  #A.update(B) =add B elements to A set  sa=set(["aa","1",2])  sb={"bb",2,3  }  sa.update(sb)  print(sa)  ###########  {2, 3, 'aa', 'bb', '1'} |
| Copy | Vs 直接assign, copy is diff id  sa=set(["aa","1",2])  sb=sa  print(sb)  print(id(sa),id(sb), sep=" ~~~~~ ")  {2, '1', 'aa'}  #####################  2325942566688 ~~~~~ 2325942566688  sc=sa.copy()  print(sc)  print(id(sc),id(sa),sep="~~~~~~~~~~")  {2, '1', 'aa'}  2325961903904~~~~~~~~~~2325942566688 |

### 9.3 set AG function

|  |
| --- |
| 1.enumerated() |
| 2.len() |
| 3.min(); max() |
| 4.sorted() |
| 5.sum() |

* Ex1:

given

|  |
| --- |
| sa={1,2,3,4}  sb=set(["python","java","c"])  sc={"AB","AA","BC",'C'} |

Test:

|  |
| --- |
| print(len(sc))  print(max(sc),min(sc),sep=" vs ")  print(sum(sa))  ################  4  C vs AA  10  #####################  Print(sum(sc))  TypeError: unsupported operand type(s) for +: 'int' and 'str' |

* Ex2:sorted

|  |
| --- |
| 1.  sa={"ban","dog","app","pig"}  print(sorted(sa)) #not change the org  print(sa) #org order not change  print(sorted(sa,reverse=True))  ['app', 'ban', 'dog', 'pig']  {'ban', 'dog', 'pig', 'app'}  ['pig', 'dog', 'ban', 'app'] |

* Ex3:numerated()

因為set 無法iteration ,as it is unordered.

S1: set to list , for loop

S2: use enumerate()

|  |
| --- |
| sa={"ban","dog","app","pig"}  for i in range(0,len(sa),1):  print(i)  0  1  2 |

S1:change to LIST

|  |
| --- |
| sa={"ban","dog","app","pig"}  sa=list(sa)  for i in range(0,len(sa),1):  print(i,sa[i],sep=" : ")  0 : ban  1 : dog  2 : pig  3 : app |

S2: enumerate()

id, val is build-in for ennumerate

|  |
| --- |
| sa={"ban","dog","app","pig"}  #sa=list(sa)  for id,val in enumerate(sa):  print(id,val) |

### 9.4 Python Set statistics

* Intro

有union ,diff, intersection

|  |  |
| --- | --- |
| Union  Days1|Days2 |  |
| Intersection  Days1&Days2 |  |
| Difference  Days1-Days2 |  |
| Symmetric Difference  c = a^b |  |

* Examples

|  |
| --- |
| .  sa=set(['aa','bb',3,4])  sb={'cc','dd',3,4}  print (type(sb),type(sa))  print(sa|sb) #union means or, use |  print(sa&sb) #intersect means and &  print(sa-sb) #show all sa elements that not exit in sb  print(sa^sb) #show sa+sb without intersection  ###############  <class 'set'> <class 'set'>  {3, 4, 'bb', 'cc', 'dd', 'aa'}  {3, 4}  {'aa', 'bb'}  {'bb', 'cc', 'dd', 'aa'} |

## Chapter 10 dictionary :

### 10.1: intro

Dictionary is used to store the data in a key-value pair format.

* Keys must be a single element
* Value can be any type such as list, tuple, integer, etc.
* Keys are unordered

|  |
| --- |
| Dict = {"Name": "Tom", "Age": 22} |

Basic extraction : Dict[key] = value.

**BUT failed to :dict[“value”]=key**

|  |
| --- |
| Create key but blank value:  my\_dict = {"Name":[],"Address":[],"Age":[]}; |

### 10.2.syntax

* 1.DML syntax

**d1={"name":"john","age":23,"title":"GT","gender":'S',16:1}**

|  |  |
| --- | --- |
| Clear(d1) |  |
| Add a new element  Ex: Da[key]=value | When key is exist, it means replace;  When key is not exist, it means add; |
| Merge other dict  Update()  Da={xxxxx}  Db={yyyyy}  Da.update(Db)  Update() | 並非修改，是直接將da.update(db) ，直接add dict\_b to da |
| Del dict | Delete a dictionary |
| Del d1[19] | Delete a key-value |

* Example:

|  |
| --- |
| d1={"name":"john","age":23,"title":"GT","gender":'S',16:1}  print(d1[16]) #return value 1  d1["title"]="MT" #replace the old value  d1[17]="t17" #add new key-value  d2={18:"t18",19:"t19"}  d1.update(d2) #merge d2 to d1  print("after add new key",d1)  d3=d1.copy()  print("d3 is copy:", d3)  ##################  1  after add new key {'name': 'john', 'age': 23, 'title': 'MT', 'gender': 'S', 16: 1, 17: 't17', 18: 't18', 19: 't19'}  d3 is copy: {'name': 'john', 'age': 23, 'title': 'MT', 'gender': 'S', 16: 1, 17: 't17', 18: 't18', 19: 't19'} |

Delete a key

|  |
| --- |
| if 19 in d1:  del d1[19]  print(d1)  {'name': 'john', 'age': 23, 'title': 'MT', 'gender': 'S', 16: 1, 17: 't17', 18: 't18'} |

* 2.manipulation syntax

|  |  |
| --- | --- |
| Len(d1) | Return a integer that num of elements |
| D1.get(key,value)  Value is optional, if key is invalid, return value  D1.get(100,”not\_exist”) | print(d1.get(100))  None  print(d1.get(100,"fill-in"))  fill-in |
| “xxx”in d1 | True/False |
| D1.items() | Return whole dictionary :key +value  With pair type:  dict\_items([('name', 'john'), ('age', 23), ('title', 'GT'), ('gender', 'S'), (16, 1)]) |
| D2=d1.copy() | Duplicate the dict |
| D1.keys()  D1.values() | Show all keys  Show all values |

|  |
| --- |
| d1={"name":"john","age":23,"title":"GT","gender":'S',16:1}  print("1 : ",len(d1))  print("2 :",d1["name"])  print("3 :",d1.get(100,"fill-in"))  print("4 ： ",d1.items()); print()  print("5 :",d1.keys(),sep=",")  print("6 : ",d1.items(),sep=" ;")  ########################  1 : 5  2 : john  3 : fill-in  4 ： dict\_items([('name', 'john'), ('age', 23), ('title', 'GT'), ('gender', 'S'), (16, 1)])  5 :,dict\_keys(['name', 'age', 'title', 'gender', 16])  6 : ;dict\_items([('name', 'john'), ('age', 23), ('title', 'GT'), ('gender', 'S'), (16, 1)]) |

### 10.3: real-application

Make a student-score input machine,

If the input name exist, return his score;

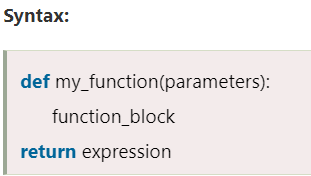
Else ask the teacher to input the score, after he type the name.

|  |
| --- |
| d4={"john":85,"peter":90,"amy":30}  name=str(input("enter st name : "))  if name in d4:  print(name,"score is %d" %d4[name])  else:  print("not exists")  score=int(input("enter his score:"))  d4[name]=score  print(name,"'s mark is: ",d4[name])  ##################  enter st name : amy  amy score is 30  ###############  enter st name : kitty  not exists  enter his score:70  kitty 's mark is: 70 |

## Chapter 11. def function

（需要配合OO, class concept ,遲下再搞）

### 11.1 intro



|  |
| --- |
| def hello\_world (): #without input parameters  print(“hello world”)  hello\_world() # function calling  ###################  hello world |

### 11.2 examples

* Example\_1:calculate area

|  |
| --- |
| def get\_area(w,h):  area=w\*h  return area  rect1=get\_area(4,6)  print(rect1)  24 |

* Tips 減少出錯

當有N 個variables, 好容易出錯，最好variables are full name

+call function 時寫明 fuc(var\_n=xxxxxx, var\_p=yyyyyy)

|  |
| --- |
| def get\_area(width,height):  area=width\*height  return area  rect1=get\_area(width=4,height=6)  print(rect1)  24 |

* Given value:

|  |
| --- |
| def get\_area(width,height=6):  area=width\*height  return area  rect1=get\_area(width=4)  print(rect1)  24 |

* Example\_2: return 2 value

|  |
| --- |
| def circle(radius):  area=(radius\*\*2)\*3.14  length=2\*radius\*3.14  return round(area,2),round(length,2)  area1=circle(5)  print(area1)  ########################  (78.5, 31.4) |

* Example\_3：Passing a List as an Argument

|  |
| --- |
| def my\_function(food):  for x in food:  print(x)  fruits = ["apple", "banana", "cherry"]  my\_function(fruits)  ######################  apple  banana  cherry |

* Example\_4:return 3 more variable

|  |
| --- |
| def test2():  return 'abc', 100, [0, 1, 2]  a, b, c = test2()  print(a)  # abc  print(b)  # 100  print(c)  # [0, 1, 2] |

## Chapter 12: CSV file

### 12.1 intro

A **csv** stands for "comma separated values", which is defined as a simple file format that uses specific structuring to arrange tabular data.

It stores tabular data such as spreadsheet or database in plain text and has a common format for data interchange.

A **csv** file opens into the excel sheet, and the rows and columns data define the standard format.

(主要分為 read(query), write, manipulation 3 parts)

（for functions: reader + writer）

### 12.2 examples

* Ex1: read txt/csv file in python

Save csv content in a text file name: test\_2.txt

Path: the test\_2.tx same as xxx.py

|  |
| --- |
| Name,Hire Date,Salary,Leaves Remaining  John Idle,08/15/14,50000.00,10  Smith Gilliam,04/07/15,65000.00,8  Parker Chapman,02/21/14,45000.00,10  Jones Palin,10/14/13,70000.00,3  Terry Gilliam,07/22/14,48000.00,7  Michael Palin,06/28/13,66000.00,8 |

**Read with pandas functions:**

|  |
| --- |
| import pandas as pd  df = pd.read\_csv('test\_2.txt')  print(df)  ########################  Name Hire Date Salary Leaves Remaining  0 John Idle 08/15/14 50000.0 10  1 Smith Gilliam 04/07/15 65000.0 8  2 Parker Chapman 02/21/14 45000.0 10  3 Jones Palin 10/14/13 70000.0 3  4 Terry Gilliam 07/22/14 48000.0 7  5 Michael Palin 06/28/13 66000.0 8 |

**Read without pandas:**

|  |
| --- |
| with open('test\_2.txt', newline='') as t2:  rows = csv.reader(t2)  for i in rows:  print(i)  ###########################################  ['Name', 'Hire Date', 'Salary', 'Leaves Remaining ']  ['John Idle', '08/15/14', '50000.00', '10 ']  ['Smith Gilliam', '04/07/15', '65000.00', '8 ']  ['Parker Chapman', '02/21/14', '45000.00', '10 ']  ['Jones Palin', '10/14/13', '70000.00', '3 ']  ['Terry Gilliam', '07/22/14', '48000.00', '7 ']  ['Michael Palin', '06/28/13', '66000.00', '8 ']  #######################################  如果直接print(rows)  <\_csv.reader object at 0x000002845983C2E0> |

* **Ex2:Write into csv with csv moduel**

No need to have a csv file before, it will auto created in the same path as xxx.py

If exists, it write in :

|  |
| --- |
| import csv  with open('test\_1.csv','w',newline='') as csvfile:  writer=csv.writer(csvfile)  writer.writerow(['name','height','weight'])  writer.writerow(['chiyu',180,72])  writer.writerow(['David',183,70]) |

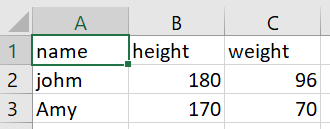
* **Ex3: write dic value in csv**

The csv module contains DictWriter method that requires name of csv file to write and a list object containing field names.

The writeheader() method writes first line in csv file as field names. The subsequent for loop writes each row in csv form to the csv file.

AS dictionary is unordered, so as long as the key-value pair are matched,

|  |
| --- |
| import csv  with open('test3.csv', 'w', newline='')as t3:  csv\_columns = ['name', 'height', 'weight'] # filednames is built-in  writer=csv.DictWriter(t3, fieldnames=csv\_columns)  writer.writeheader()  writer.writerow({'name':'johm','weight':96,'height':180})  writer.writerow({'name':'Amy', 'weight':70,'height':170}) |



Another example from web:

可以一次個將data assign to a dictionary, then can write in 2 ways:

S1:writerow+ for loop(data)

S2:writerows(dict\_data)

|  |
| --- |
| import csv  csv\_columns = ['No','Name','Country']  dict\_data = [  {'No': 1, 'Name': 'Alex', 'Country': 'India'},  {'No': 2, 'Name': 'Ben', 'Country': 'USA'},  {'No': 3, 'Name': 'Shri Ram', 'Country': 'India'},  {'No': 4, 'Name': 'Smith', 'Country': 'USA'},  {'No': 5, 'Name': 'Yuva Raj', 'Country': 'India'},  ]  csv\_file = "Names.csv"  try:      with open(csv\_file, 'w') as csvfile:          writer = csv.DictWriter(csvfile, fieldnames=csv\_columns)          writer.writeheader()          for data in dict\_data:              writer.writerow(data)  ##writer.writerows(dict\_data)  except IOError:      print("I/O error") |

* Ex\_5: write in with pandas

It is as easy as reading the CSV file using pandas. You need to create the DataFrame, which is a two-dimensional, heterogeneous tabular data structure and consists of three main components- data, columns, and rows.

Here, we take a slightly more complicated file to read, called hrdata.csv, which contains data of company employees.

### 12.3 concept:

(目前不用理)

The CSV module work is to handle the CSV files to read/write and get data from specified columns. There are different types of CSV functions, which are as follows:

* **csv.field\_size\_limit -** It returns the current maximum field size allowed by the parser.
* **csv.get\_dialect -** Returns the dialect associated with a name.
* **csv.list\_dialects -** Returns the names of all registered dialects.
* **csv.reader -** Read the data from a CSV file
* **csv.register\_dialect -** It associates dialect with a name, and name must be a string or a Unicode object.
* **csv.writer -** Write the data to a CSV file
* **csv.unregister\_dialect -** It deletes the dialect, which is associated with the name from the dialect registry. If a name is not a registered dialect name, then an error is being raised.
* **csv.QUOTE\_ALL -** It instructs the writer objects to quote all fields.
* **csv.QUOTE\_MINIMAL -** It instructs the writer objects to quote only those fields which contain special characters such as quotechar, delimiter, etc.
* **csv.QUOTE\_NONNUMERIC -** It instructs the writer objects to quote all the non-numeric fields.
* **csv.QUOTE\_NONE -** It instructs the writer object never to quote the fields.
* Writer function：

We can also write any new and existing CSV files in Python by using the csv.writer() module.

It is similar to the csv.reader() module and also has two methods, i.e., writer function or the Dict Writer class.

It presents two functions, i.e., writerow() and writerows(). The writerow() function only write one row, and the writerows() function write more than one row.

It is defined as a construct that allows you to create, store, and re-use various formatting parameters. It supports several attributes; the most frequently used are:

* **Dialect.delimiter:** This attribute is used as the separating character between the fields. The default value is a comma (,).
* **Dialect.quotechar:** This attribute is used to quote fields that contain special characters.
* **Dialect.lineterminator:** It is used to create new lines, and the default value is '\r\n'.

## Chapter 13. excel with pandas

(Just focus on use pandas to read/ write excel)

### 13.1 intro

(**excel= a workbook consist of many spreadsheets.)**

An Excel spreadsheet document is called a workbook which is saved in a file with **.xlsx extension**.

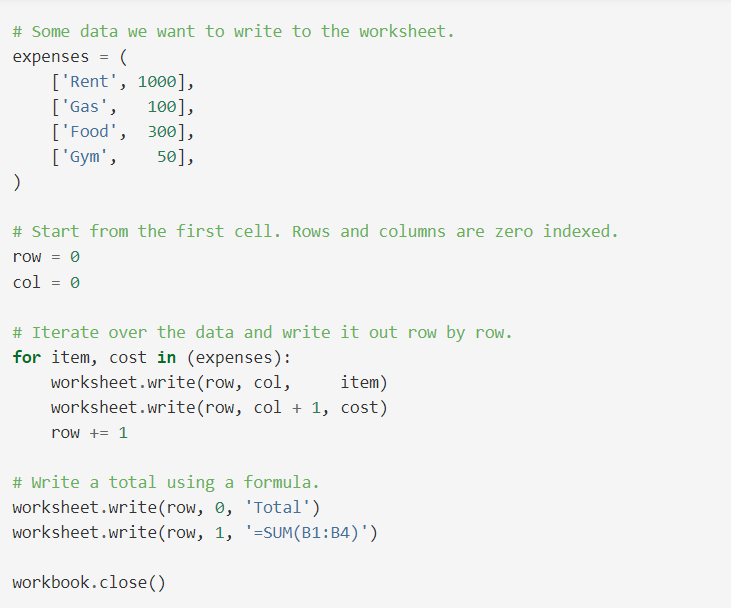
The first row of the spreadsheet is mainly reserved for the header, while the first column identifies the sampling unit.

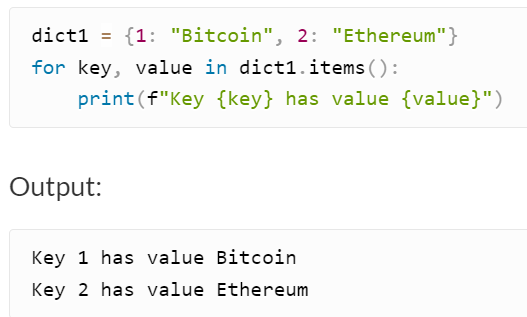
Each workbook can contain multiple sheets that are also called a worksheets.

A box at a particular column and row is called a cell, and each cell can include a number or text value. The grid of cells with data forms a sheet.

### 13.1 create a workbook with Py

**(這裡需要學識 mulit-variable for loop, 用於：dictionary & list)**







* Ex1: insert cells

|  |
| --- |
| import openpyxl  wb=openpyxl.Workbook() #Workbook() is class  sheet=wb.worksheets[0] #worksheet/wb are variables  #sheet['A1']='hello'  listtile=["name",'phe\_no']  sheet.append(listtile)  wb.save('ex\_test1.xlsx') |

### 13.2 read excel with pandas

|  |
| --- |
| import pandas as pd  df = pd.read\_excel('users.xlsx', sheet\_name = [0,1,2])  df = pd.read\_excel('users.xlsx', sheet\_name = ['User\_info','compound'])  df = pd.read\_excel('users.xlsx', sheet\_name = None) # read all sheets |

* Example:1

**Pd=pd.read\_excel(path\<name>,sheent\_name=[‘xxxxxxxxx’])**

|  |
| --- |
| import pandas as pd  df = pd.read\_excel (r'C:\Users\fengs\Desktop\py\_project\csv\_file\ex\_test1.xlsx'  ,  sheet\_name=['s2']  )  #place "r" before the path string to address special character,  #such as '\'. Don't forget to put the file name at the end of the path + '.xlsx'  print (df) |
| {'s2': Product Price  0 Desktop Computer 700  1 Tablet 250  2 Printer 120  3 Laptop 1200} |

## Chapter 14:Pandas

### 1.intro:

The two main data structures in Pandas are series and dataframe.

Datafame is similar with excel

* Data frame :Parameter & Description:

**data:** It consists of different forms like ndarray, series, map, constants, lists, array.

**index:** The Default np.arrange(n) index is used for the row labels if no index is passed.

**columns:** The default syntax is np.arrange(n) for the column labels. It shows only true if no index is passed.

**dtype:** It refers to the data type of each column.

**copy():** It is used for copying the data.

### 14.2.pandas

* Ex1 empty dataframe:

|  |
| --- |
| import pandas as pd  df = pd.DataFrame()  print (df)  #####################  Empty DataFrame  Columns: []  Index: [] |

* Ex2: basic dataframe

**No index=[],columns=[]**

|  |
| --- |
| x=['john','peter']  df=pd.DataFrame(x)  print(x)  print(df)  ###############  ['john', 'peter']  0  0 john  1 peter |

### 14.3Write-in list data :

2-d matrix, self-made : index=[‘xxx’]; columns=[‘yyyyy’]

Lists外面要被一個[] 包住

|  |
| --- |
| df=pd.DataFrame(  [  [60,61,63,67],  [70,12,66,34],  [85,92,86,32],  [90,32,98,43]  ],  index=['peter','john','amy','kitty'],  columns=['math','chin','Eng','IS']  )  print(df)  print(df['math'])  #####################  math chin Eng IS  peter 60 61 63 67  john 70 12 66 34  amy 85 92 86 32  kitty 90 32 98 43 |

### 14.4 (重點)dict of nD list/arrays

**+select column : dict[‘index\_name’]**

In the above below, we have defined a dictionary named "info" that consist list of ID and Department. For printing the values, we have to call the info dictionary through a variadble called df and pass it as an argument in print().

|  |
| --- |
| import pandas as pd  info = {'ID' :[101, 102, 103],'Department' :['B.Sc','B.Tech','M.Tech',]}  df = pd.DataFrame(info)  print (df)  print(df[‘ID’])  #######################  ID Department  0 101 B.Sc  1 102 B.Tech  2 103 M.Tech |

### 14.5： series

Note: change index to abcd

|  |
| --- |
| import pandas as pd  info = {'one' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f']),  'two' : pd.Series([1, 2, 3, 4, 5, 6, 7, 8], index=['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h'])}  d1 = pd.DataFrame(info)  print (d1)  print(d1[2:5])  #############################  one two  a 1.0 1  b 2.0 2  c 3.0 3  d 4.0 4  e 5.0 5  f 6.0 6  g NaN 7  h NaN 8 |

* Add column

|  |
| --- |
| df['three']=pd.Series([20,40,60],index=['a','b','c'])  df['four']=df['one']+df['three']  // |

### 15.6:row syntax

**Given: dict[‘<index\_name>’] is column selection**

select any row by passing the row label to a **loc** function.

* Row select:loc/iloc

loc=location

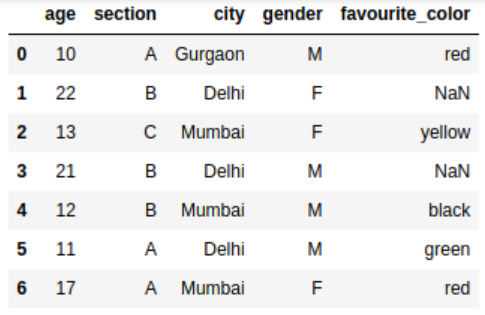
iloc=integer location

loc is label-based, which means that we have to specify the name of the rows and columns that we need to filter out.

iloc is integer index-based. So here, we have to specify rows and columns by their integer index.

Given:

|  |
| --- |
| data = pd.DataFrame({  'age' : [ 10, 22, 13, 21, 12, 11, 17],  'section' : [ 'A', 'B', 'C', 'B', 'B', 'A', 'A'],  'city' : [ 'Gurgaon', 'Delhi', 'Mumbai', 'Delhi', 'Mumbai', 'Delhi', 'Mumbai'],  'gender' : [ 'M', 'F', 'F', 'M', 'M', 'M', 'F'],  'favourite\_color' : [ 'red', np.NAN, 'yellow', np.NAN, 'black', 'green', 'red']  })  print(data)  print(data[2:5]) # if not use loc/iloc  #######################################  age section city gender favourite\_color  0 10 A Gurgaon M red  1 22 B Delhi F NaN  2 13 C Mumbai F yellow  3 21 B Delhi M NaN  4 12 B Mumbai M black  5 11 A Delhi M green  6 17 A Mumbai F red |



* Cases

<df\_name>.loc[ [cond1]&[cond2]…..]

|  |
| --- |
| 1.print(data[2:5]) #print row 2-4  2.print(data.loc[data.age>15]) # where age>15  3. data.loc[(data.age >= 12) & (data.gender == 'M')]  4. data.loc[(data.age >= 12), ['city', 'gender']] # select 2 colums:city,gender  #########################  5. Update the values of a particular column on selected rows  # update a column with condition  data.loc[(data.age >= 12), ['section']] = 'M'  ##############  # select a range of rows and columns  data.iloc[1:3,2:4] |

* Iloc

dataframe.iloc[row, column]

|  |
| --- |
| 1.# select a range of rows and columns  data.iloc[1:3,2:4]  2. Return the the value of the second [1] row of the first [0] column:  print(df.iloc[1, 0])  3.many but not range  df.iloc[[0, 2], [0, 1]] |

* Head()/tail()

Df.head[2:5] ：error

|  |
| --- |
| Df.tail(2) #最尾2 行 |
| Head(5) #頭5行  Or df,head(n=5) |

### 15.7: manipulation syntax

* 1. Table list

|  |  |
| --- | --- |
| Functions | info |
| [Pandas DataFrame.append()](https://www.javatpoint.com/pandas-append) | Add the rows of other dataframe to the end of the given dataframe. |
| [Pandas DataFrame.apply()](https://www.javatpoint.com/pandas-apply) | Allows the user to pass a function and apply it to every single value of the Pandas series. |
| [Pandas DataFrame.assign()](https://www.javatpoint.com/pandas-dataframe-assign) | Add new column into a dataframe. |
| [Pandas DataFrame.astype()](https://www.javatpoint.com/pandas-dataframe-astype) | Cast the Pandas object to a specified dtype.astype() function. |
| [Pandas DataFrame.concat()](https://www.javatpoint.com/pandas-concatenation) | Perform concatenation operation along an axis in the DataFrame. |
| [Pandas DataFrame.count()](https://www.javatpoint.com/pandas-count) | Count the number of non-NA cells for each column or row. |
| [Pandas DataFrame.describe()](https://www.javatpoint.com/pandas-dataframe-describe) | Calculate some statistical data like percentile, mean and std of the numerical values of the Series or DataFrame. |
| [Pandas DataFrame.drop\_duplicates()](https://www.javatpoint.com/pandas-dataframe-drop_duplicates) | Remove duplicate values from the DataFrame. |
| [Pandas DataFrame.groupby()](https://www.javatpoint.com/pandas-groupby) | Split the data into various groups. |
| [Pandas DataFrame.head()](https://www.javatpoint.com/pandas-dataframe-head) | Returns the first n rows for the object based on position. |
| [Pandas DataFrame.hist()](https://www.javatpoint.com/pandas-dataframe-hist) | Divide the values within a numerical variable into "bins". |
| [Pandas DataFrame.iterrows()](https://www.javatpoint.com/pandas-dataframe-iterrows) | Iterate over the rows as (index, series) pairs. |
| [Pandas DataFrame.mean()](https://www.javatpoint.com/pandas-dataframe-mean) | Return the mean of the values for the requested axis. |
| [Pandas DataFrame.melt()](https://www.javatpoint.com/pandas-melt) | Unpivots the DataFrame from a wide format to a long format. |
| [Pandas DataFrame.merge()](https://www.javatpoint.com/pandas-merge) | Merge the two datasets together into one. |
| [Pandas DataFrame.pivot\_table()](https://www.javatpoint.com/pandas-pivot-table) | Aggregate data with calculations such as Sum, Count, Average, Max, and Min. |
| [Pandas DataFrame.query()](https://www.javatpoint.com/pandas-dataframe-query) | Filter the dataframe. |
| [Pandas DataFrame.sample()](https://www.javatpoint.com/pandas-dataframe-sample) | Select the rows and columns from the dataframe randomly. |
| [Pandas DataFrame.shift()](https://www.javatpoint.com/pandas-shift) | Shift column or subtract the column value with the previous row value from the dataframe. |
| [Pandas DataFrame.sort()](https://www.javatpoint.com/python-pandas-sorting) | Sort the dataframe. |
| [Pandas DataFrame.sum()](https://www.javatpoint.com/pandas-sum) | Return the sum of the values for the requested axis by the user. |
| [Pandas DataFrame.to\_excel()](https://www.javatpoint.com/pandas-dataframe-to_excel) | Export the dataframe to the excel file. |
| [Pandas DataFrame.transpose()](https://www.javatpoint.com/pandas-dataframe-transpose) | Transpose the index and columns of the dataframe. |
| [Pandas DataFrame.where()](https://www.javatpoint.com/pandas-dataframe-where) | Check the dataframe for one or more conditions. |

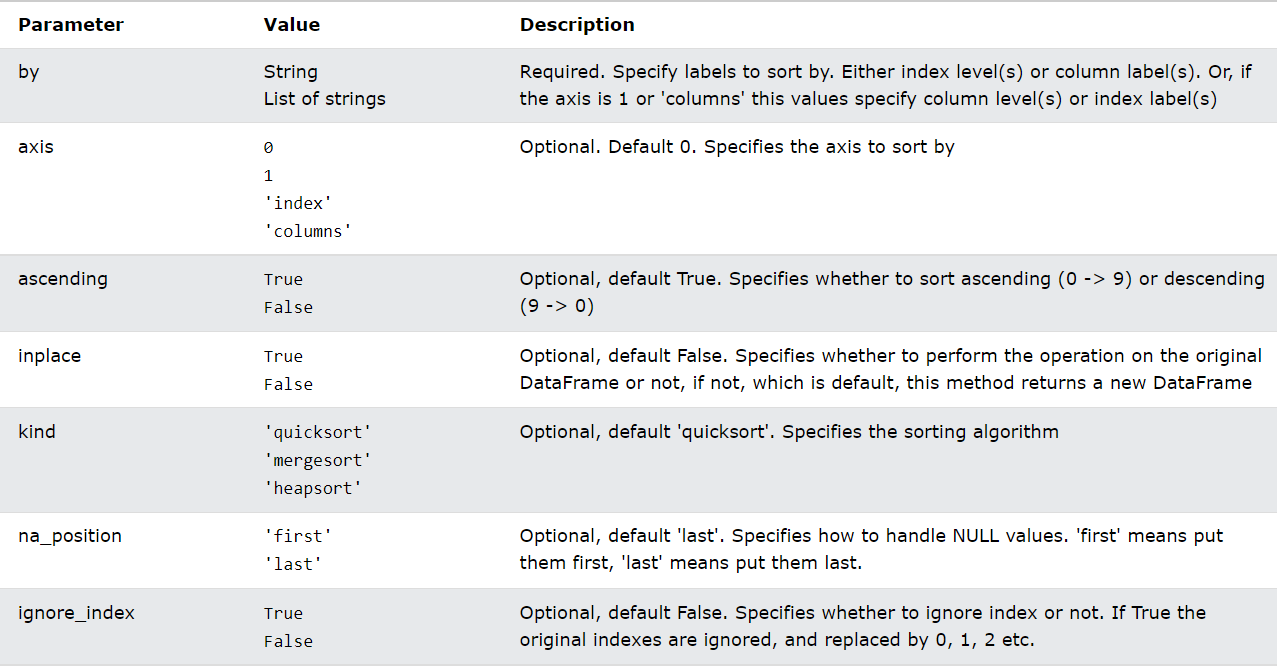
* Ex2:sort\_values()

通常用

1：ascencding=True/False

2：na\_position=’first’/’last’ ####null 放頭或尾巴

|  |
| --- |
| *dataframe*.sort\_values(by, axis, ascending, inplace, kind, na\_position, ignore\_index, key) |



|  |
| --- |
| data.sort\_values(by='age',ascending=False) #False:descending  **data.sort\_values(by=[‘age’,’city’]]) #sorted by 2 column**  ###############################  age section city gender favourite\_color  1 22 B Delhi F NaN  3 21 B Delhi M NaN  6 17 A Mumbai F red  2 13 C Mumbai F yellow  4 12 B Mumbai M black  5 11 A Delhi M green  0 10 A Gurgaon M red |

* Ex2: DataFrame.describe()

計算median, 頭尾25%，75%之類：

|  |
| --- |
| DataFrame.describe(percentiles=None, include=None, exclude=None) |

The describe() method is used for calculating some statistical data like **percentile, mean** and **std** of the numerical values of the Series or DataFrame.

It analyzes both numeric and object series and also the DataFrame column sets of mixed data types.

* **percentile:** It is an optional parameter which is a list like data type of numbers that should fall between 0 and 1. Its default value is [.25, .5, .75], which returns the 25th, 50th, and 75th percentiles.
* **include:** It is also an optional parameter that includes the list of the data types while describing the DataFrame. Its default value is None.
* **exclude:** It is also an optional parameter that exclude the list of data types while describing DataFrame. Its default value is None.

### 15.8 pandas CSV read data

* 1.read csv

|  |
| --- |
| Pandas.read\_csv(file\_name,[header=,index\_col,encoding=,sep=]) |

|  |
| --- |
| pandas.read\_csv(filepath\_or\_buffer, sep=', ', delimiter=None,  header='infer', names=None,  index\_col=None, usecols=None, squeeze=False,  ..., engine=None, ...) |

Note1：path: as the py file as csv file are not in a same folder, so need to add a full

Path

Note2: remember to add .csv after the filed name

Note3: C error: EOF inside string starting at line 312074 , so need to add engine=’python’, as the rows are too many.

Note4: ParserError: unexpected end of data, so add error\_bad\_lines=False

|  |
| --- |
| import pandas as pd  data=pd.read\_csv(r'C:\Users\fengs\Desktop\py\_project\csv\_file\mlb\_players.csv',\  engine='python',error\_bad\_lines=False)  print(data.head(n=6)) #print()  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  print(data.head(n=6))  Name "Team" ... "Weight(lbs)" "Age"  0 Adam Donachie "BAL" ... 180 22.99  1 Paul Bako "BAL" ... 215 34.69  2 Ramon Hernandez "BAL" ... 210 30.78  3 Kevin Millar "BAL" ... 210 35.43  4 Chris Gomez "BAL" ... 188 35.71  5 Brian Roberts "BAL" ... 176 29.39 |

* Ex:2.Iloc syntax +loc basic

|  |
| --- |
| data.iloc[0:5] # first five rows of dataframe  data.iloc[:, 0:2] # first two columns of data frame with all rows  data.iloc[[0,3,6,24], [0,5,6]] # 1st, 4th, 7th, 25th row + 1st 6th 7th columns.  data.iloc[0:5, 5:8] # first 5 rows and 5th, 6th, 7th columns of data frame (county -> phone1). |

|  |
| --- |
| print(data.iloc[[0,3,6,24], [0,3,5,4]])  print(data.loc[2:10],['Name','Age','Team'])  #############################  Name "Height(inches)" Age "Weight(lbs)"  0 Adam Donachie 74 22.99 180  3 Kevin Millar 72 35.43 210  6 Miguel Tejada 69 30.77 209  24 Scott Williamson 72 31.03 180 |

* Ex3: describe method()

## Common error

### 1. 'str' object is not callable

In a script, has define str variable, so variable replace build-in str function.

Solution: del str

### 2. 'set' object is not subscriptable

s1=set([1,2,'CC',"dd","EGG"])

print(s1[2])

set data structure is referred as Unordered Collections of Unique Elements and that doesn't support operations like indexing or slicing

### 3.scv write in error

[Errno 13] Permission denied: 'test\_1.csv’

呢個file 被人開緊，或者沒有開啟權限

### 4.pandas: \_\_init\_\_() got multiple values for argument 'index'

幾條list values 要被一個[ ] 括住，當成一個value