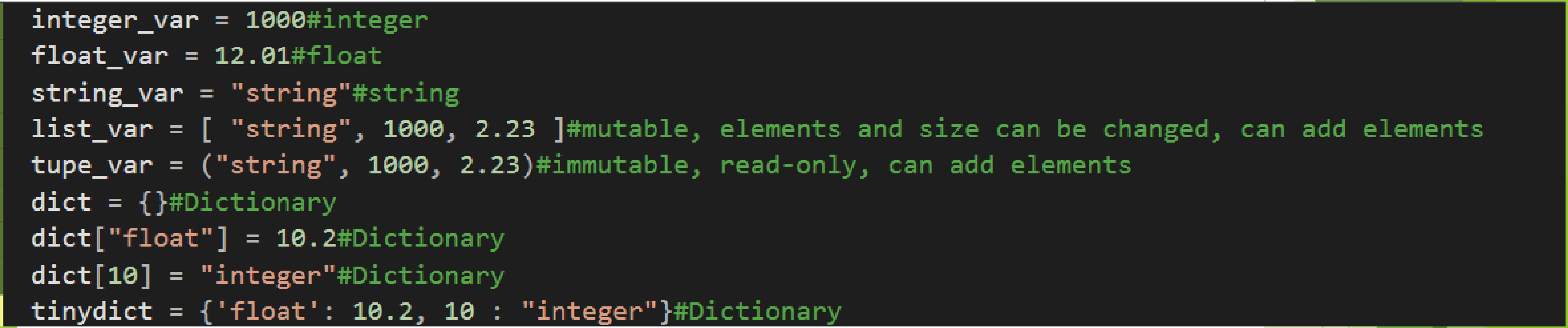
Python

Outline

- Basic Variable Type
- Basic Operators
- Decision Making
- Loops
- String
- LIST
- Tuple
- Dictionary
- File I/O
- Object Oriented

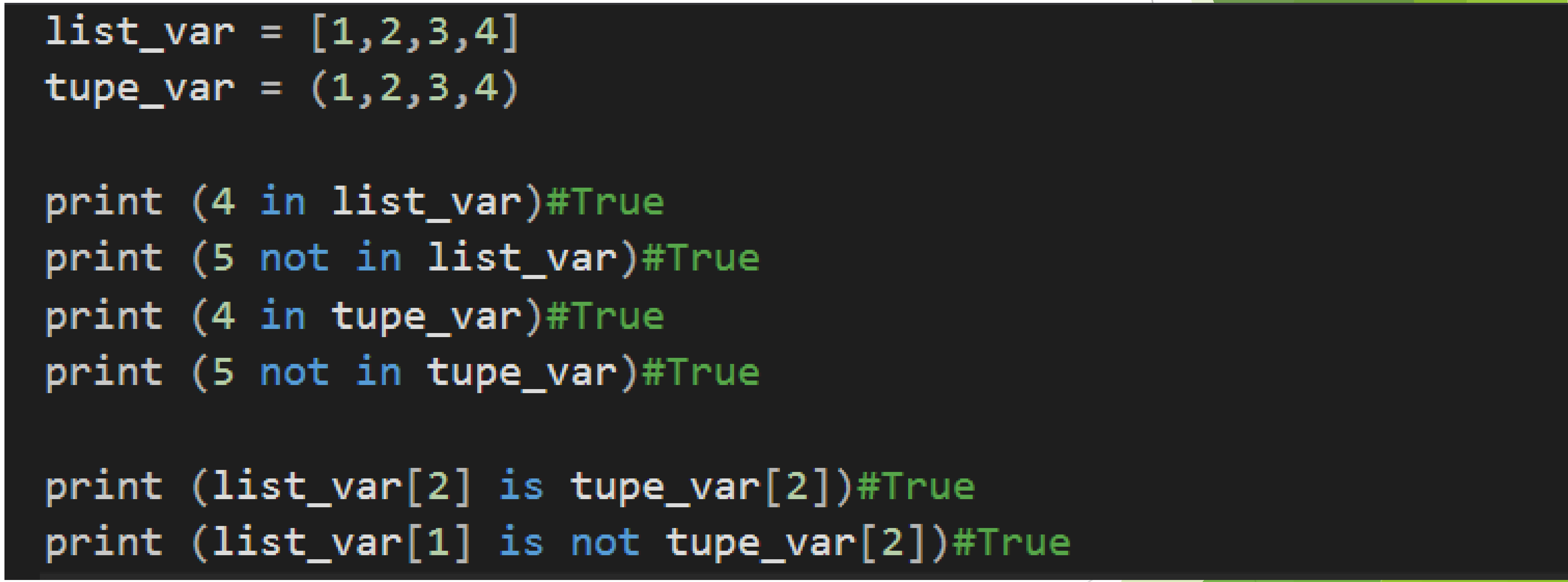
Basic Variable Type



```
integer_var = 1000#integer
float_var = 12.01#float
string_var = "string"#string
list_var = [ "string", 1000, 2.23 ]#mutable, elements and size can be changed, can add elements
tupe_var = ("string", 1000, 2.23)#immutable, read-only, can add elements
dict = {}#Dictionary
dict["float"] = 10.2#Dictionary
dict[10] = "integer"#Dictionary
tinydict = {'float': 10.2, 10 : "integer"}#Dictionary
```

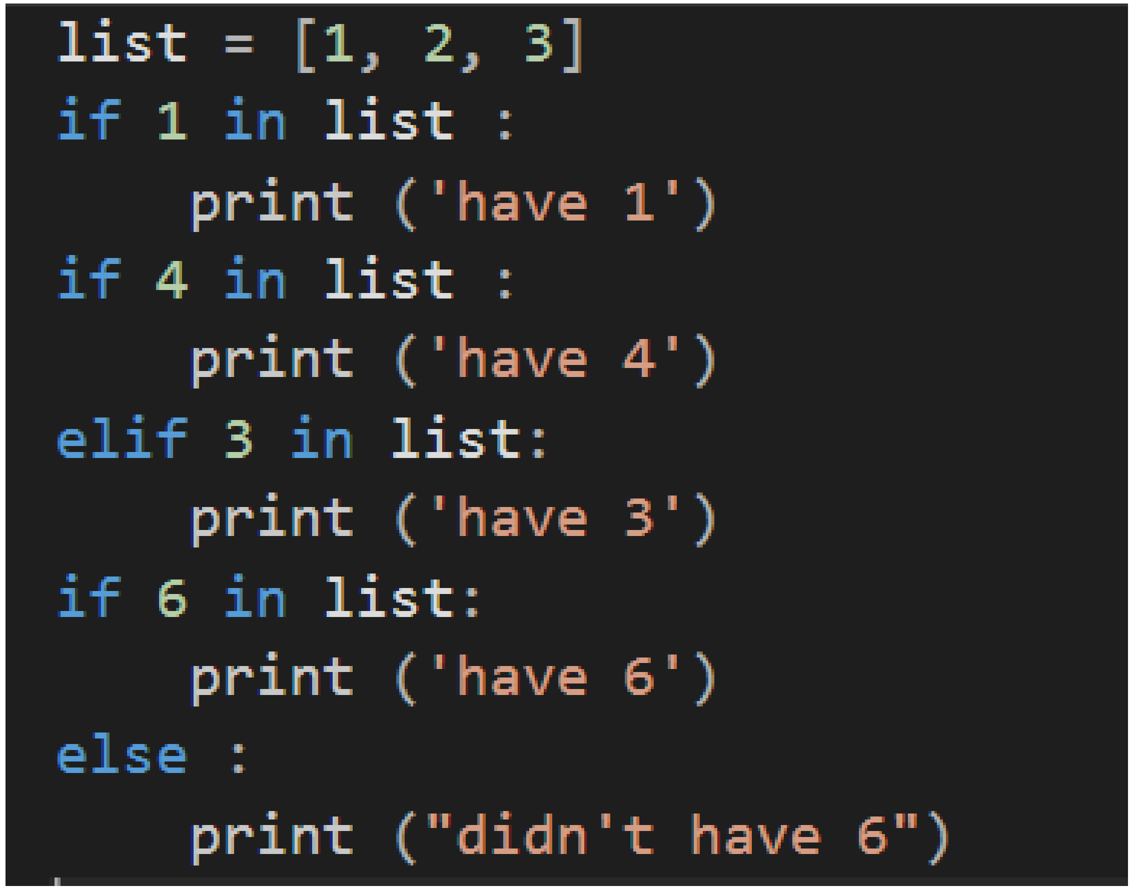
Basic Operators

- + : Addition
- - : Subtraction
- * : Multiplication
- / : Division
- ** : Exponent

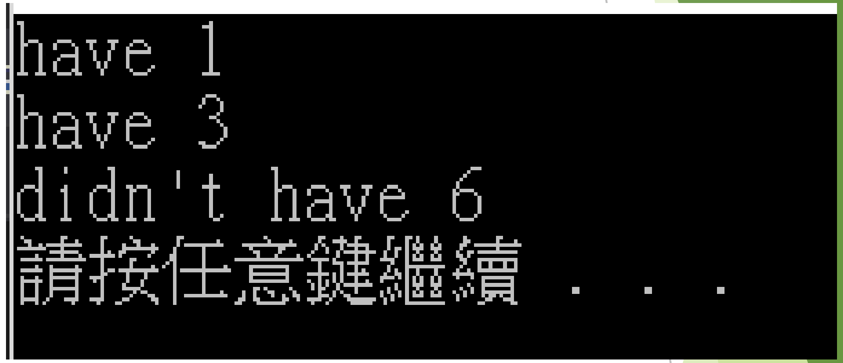


```
list_var = [1,2,3,4]
tupe_var = (1,2,3,4)

print (4 in list_var)#True
print (5 not in list_var)#True
print (4 in tupe_var)#True
print (5 not in tupe_var)#True

print (list_var[2] is tupe_var[2])#True
print (list_var[1] is not tupe_var[2])#True
```

Decision Making

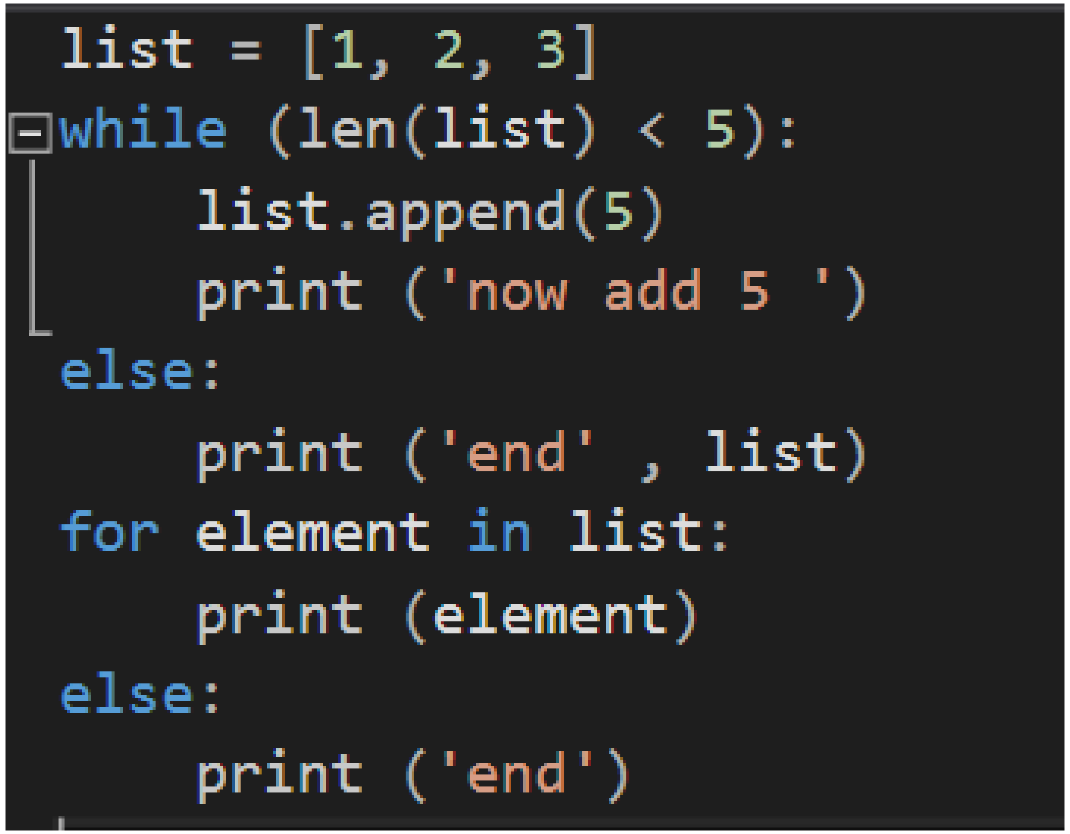


```
list = [1, 2, 3]
if 1 in list :
    print ('have 1')
if 4 in list :
    print ('have 4')
elif 3 in list:
    print ('have 3')
if 6 in list:
    print ('have 6')
else :
    print ("didn't have 6")
```

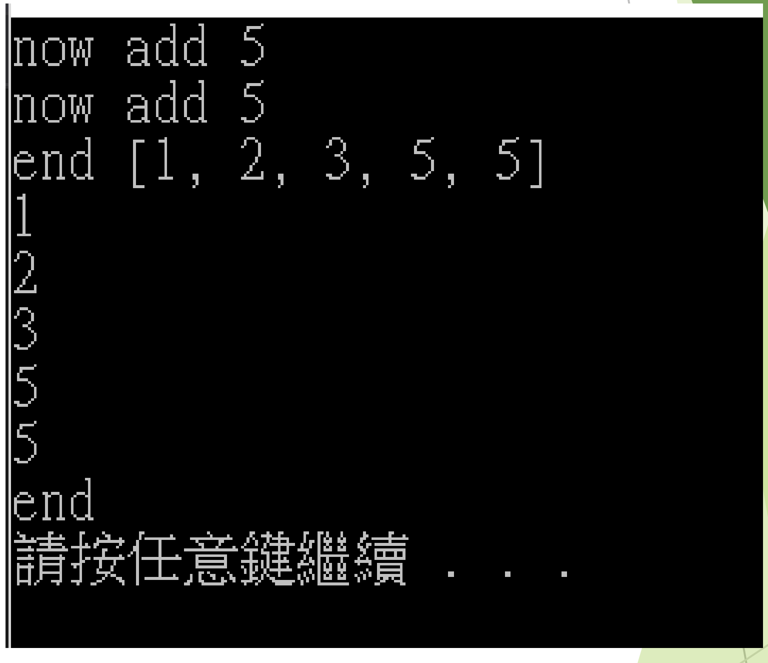


```
have 1
have 3
didn't have 6
請按任意鍵繼續 . . .
```

Loops

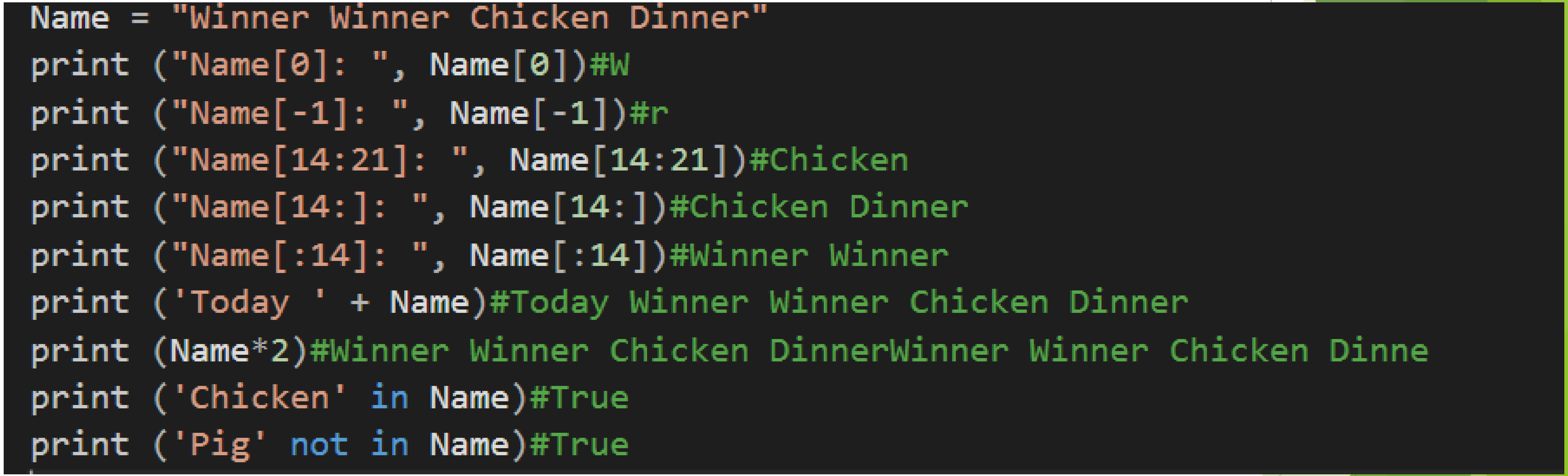


```
list = [1, 2, 3]
while (len(list) < 5):
    list.append(5)
    print ('now add 5 ')
else:
    print ('end' , list)
for element in list:
    print (element)
else:
    print ('end')
```



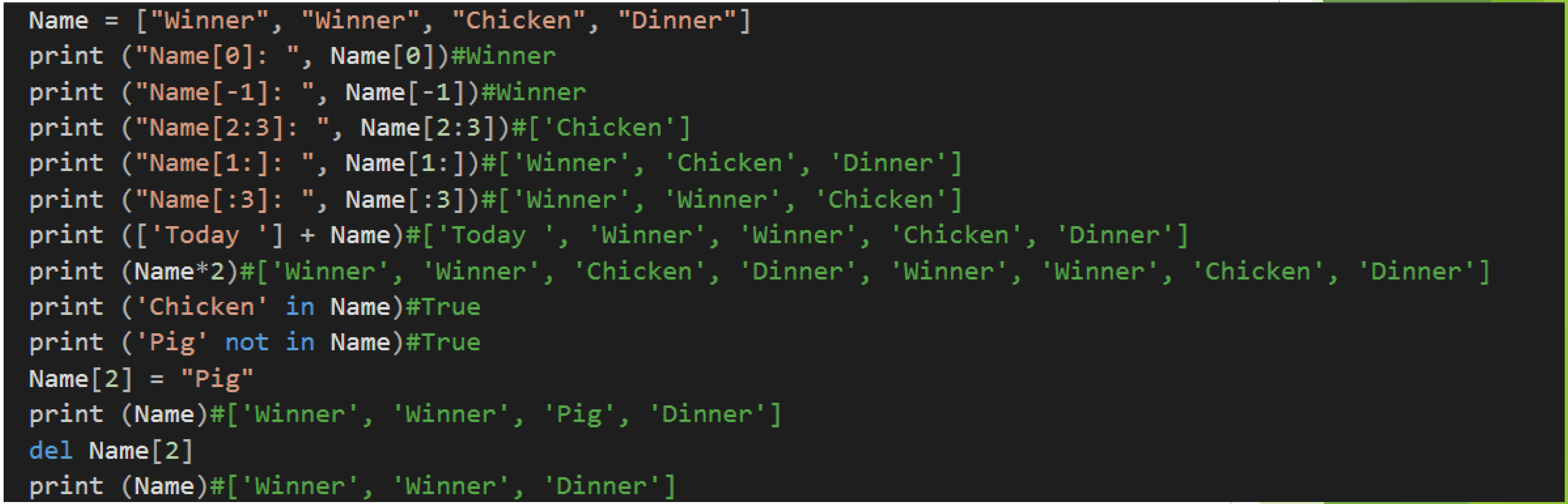
```
now add 5
now add 5
end [1, 2, 3, 5, 5]
1
2
3
5
5
end
請按任意鍵繼續 . . .
```

String



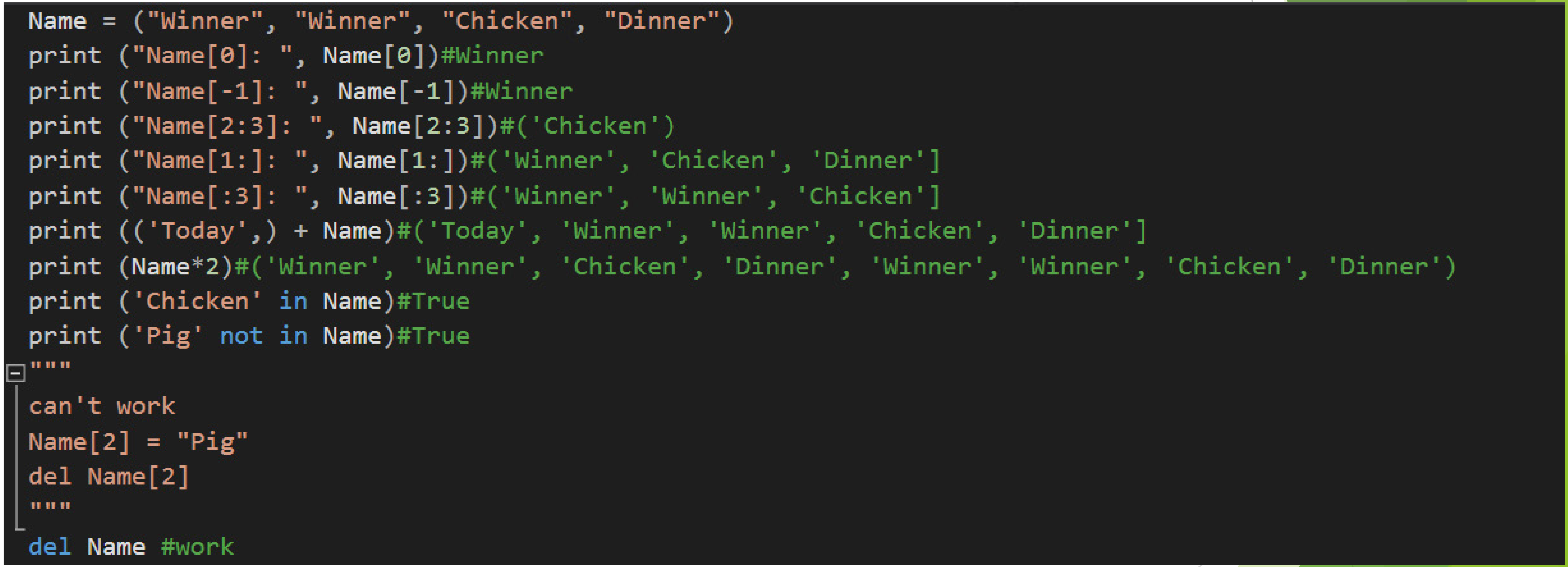
```
Name = "Winner Winner Chicken Dinner"
print ("Name[0]: ", Name[0])#W
print ("Name[-1]: ", Name[-1])#r
print ("Name[14:21]: ", Name[14:21])#Chicken
print ("Name[14:]: ", Name[14:])#Chicken Dinner
print ("Name[:14]: ", Name[:14])#Winner Winner
print ('Today ' + Name)#Today Winner Winner Chicken Dinner
print (Name*2)#Winner Winner Chicken DinnerWinner Winner Chicken Dinne
print ('Chicken' in Name)#True
print ('Pig' not in Name)#True
```

List



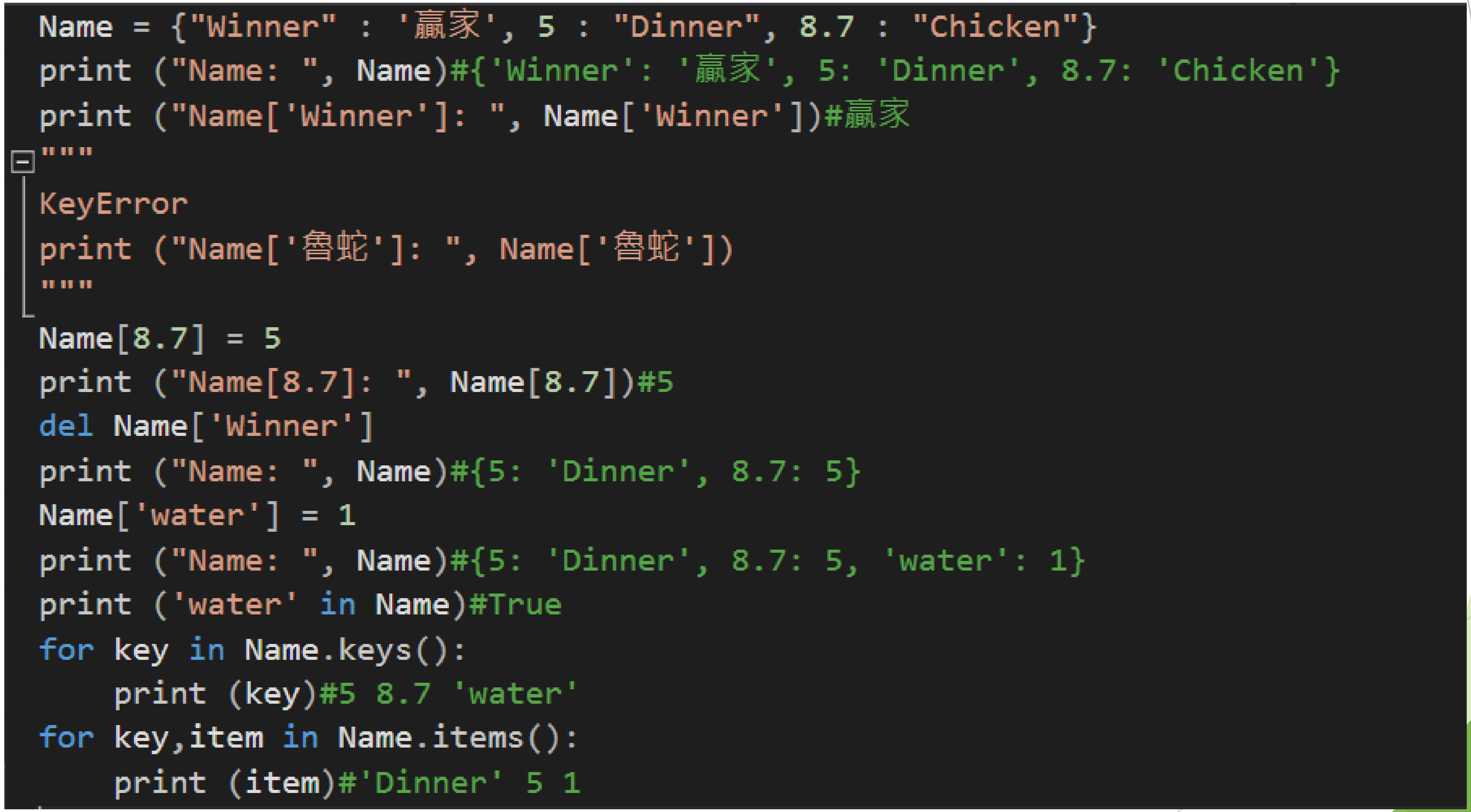
```python
Name = ["Winner", "Winner", "Chicken", "Dinner"]
print ("Name[0]: ", Name[0])#Winner
print ("Name[-1]: ", Name[-1])#Winner
print ("Name[2:3]: ", Name[2:3])#['Chicken']
print ("Name[1:]: ", Name[1:])#['Winner', 'Chicken', 'Dinner']
print ("Name[:3]: ", Name[:3])#['Winner', 'Winner', 'Chicken']
print (['Today '] + Name)#['Today ', 'Winner', 'Winner', 'Chicken', 'Dinner']
print (Name*2)#['Winner', 'Winner', 'Chicken', 'Dinner', 'Winner', 'Winner', 'Chicken', 'Dinner']
print ('Chicken' in Name)#True
print ('Pig' not in Name)#True
Name[2] = "Pig"
print (Name)#['Winner', 'Winner', 'Pig', 'Dinner']
del Name[2]
print (Name)#['Winner', 'Winner', 'Dinner']
```

Tuple



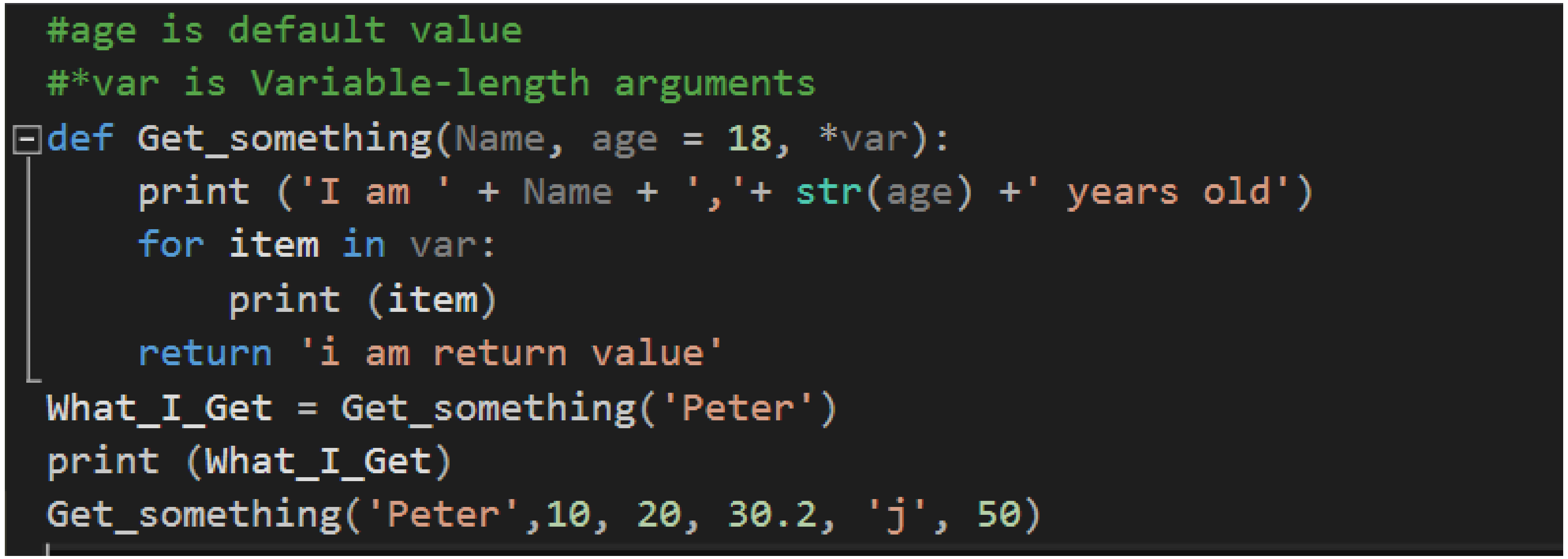
```python
Name = ("Winner", "Winner", "Chicken", "Dinner")
print ("Name[0]: ", Name[0])#Winner
print ("Name[-1]: ", Name[-1])#Winner
print ("Name[2:3]: ", Name[2:3])#('Chicken')
print ("Name[1:]: ", Name[1:])#('Winner', 'Chicken', 'Dinner']
print ("Name[:3]: ", Name[:3])#('Winner', 'Winner', 'Chicken']
print (('Today',) + Name)#('Today', 'Winner', 'Winner', 'Chicken', 'Dinner']
print (Name*2)#('Winner', 'Winner', 'Chicken', 'Dinner', 'Winner', 'Winner', 'Chicken', 'Dinner')
print ('Chicken' in Name)#True
print ('Pig' not in Name)#True
"""
can't work
Name[2] = "Pig"
del Name[2]
"""
del Name #work
```

Dictionary

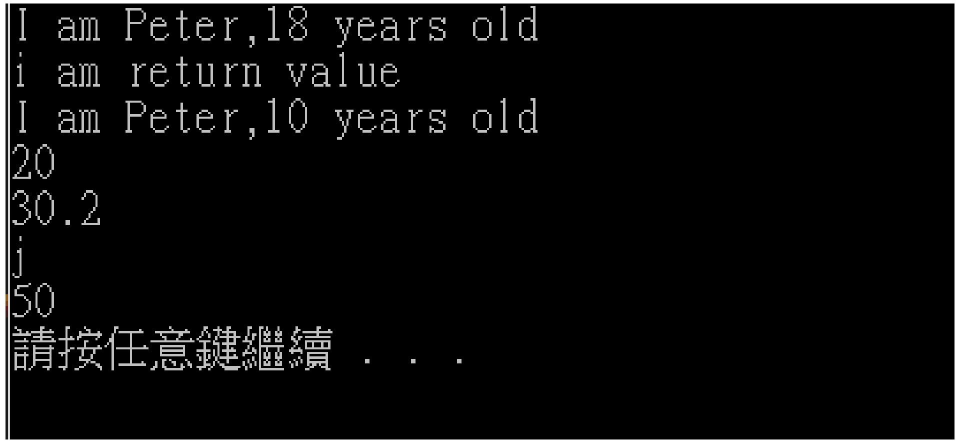


```
Name = {"Winner" : '贏家', 5 : "Dinner", 8.7 : "Chicken"}
print ("Name: ", Name)#{'Winner': '贏家', 5: 'Dinner', 8.7: 'Chicken'}
print ("Name['Winner']: ", Name['Winner'])#贏家
"""
KeyError
print ("Name['魯蛇']: ", Name['魯蛇'])
"""
Name[8.7] = 5
print ("Name[8.7]: ", Name[8.7])#5
del Name['Winner']
print ("Name: ", Name)#{5: 'Dinner', 8.7: 5}
Name['water'] = 1
print ("Name: ", Name)#{5: 'Dinner', 8.7: 5, 'water': 1}
print ('water' in Name)#True
for key in Name.keys():
    print (key)#5 8.7 'water'
for key,item in Name.items():
    print (item)#'Dinner' 5 1
```

Function



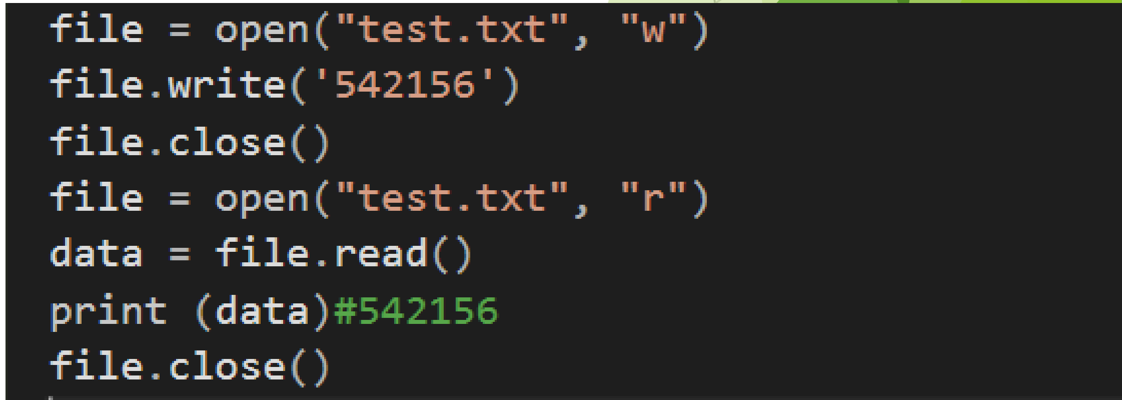
```
#age is default value
#*var is Variable-length arguments
def Get_something(Name, age = 18, *var):
    print ('I am ' + Name + ','+ str(age) +' years old')
    for item in var:
        print (item)
    return 'i am return value'
What_I_Get = Get_something('Peter')
print (What_I_Get)
Get_something('Peter',10, 20, 30.2, 'j', 50)
```



```
I am Peter,18 years old
i am return value
I am Peter,10 years old
20
30.2
j
50
請按任意鍵繼續 . . .
```

File I/O

- r : read only
- rb : read only in binary format
- r+ : read and write, doesn't delete the content of the file, doesn't create a new file if file doesn't exist, if read before write, it will write for appending
- rb+ : r+ in binary format
- r : write only
- wb : write only in binary format
- w+ : read and write, deletes the content of the file , creates it if it doesn't exist
- wb+ : w+ in binary format
- a : open file for appending, creates it if it doesn't exist
- ab : a in binary format
- a+ : read and write for appending
- ab+ : read and write for appending in binary format

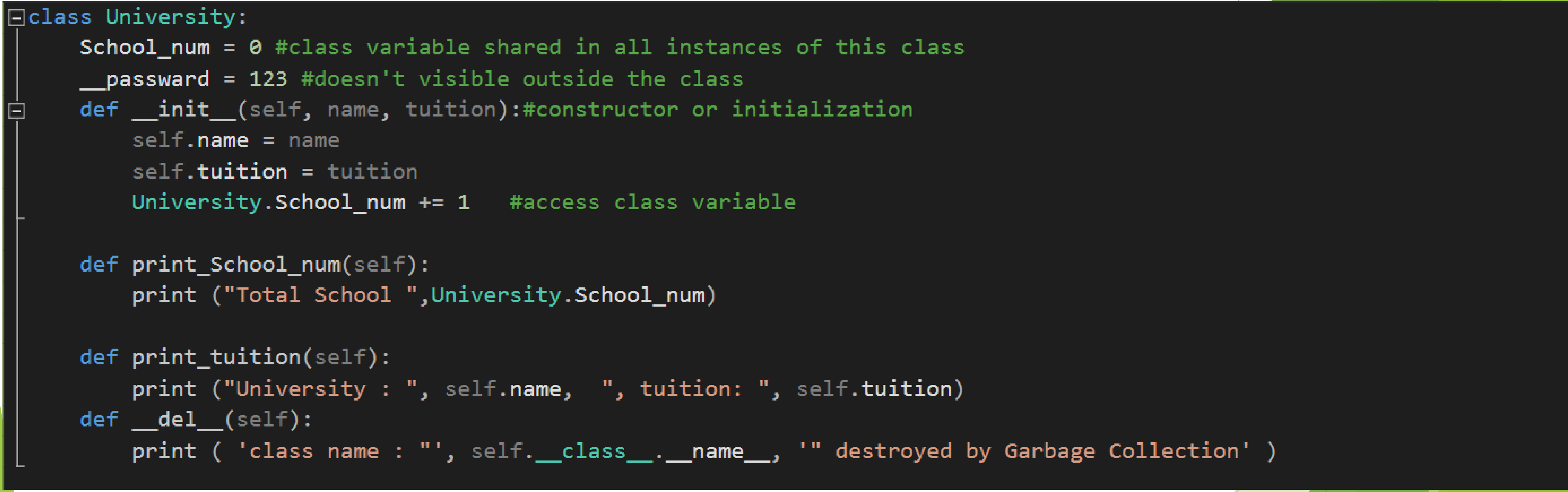


```
file = open("test.txt", "w")
file.write('542156')
file.close()
file = open("test.txt", "r")
data = file.read()
print (data)#542156
file.close()
```

Object Oriented

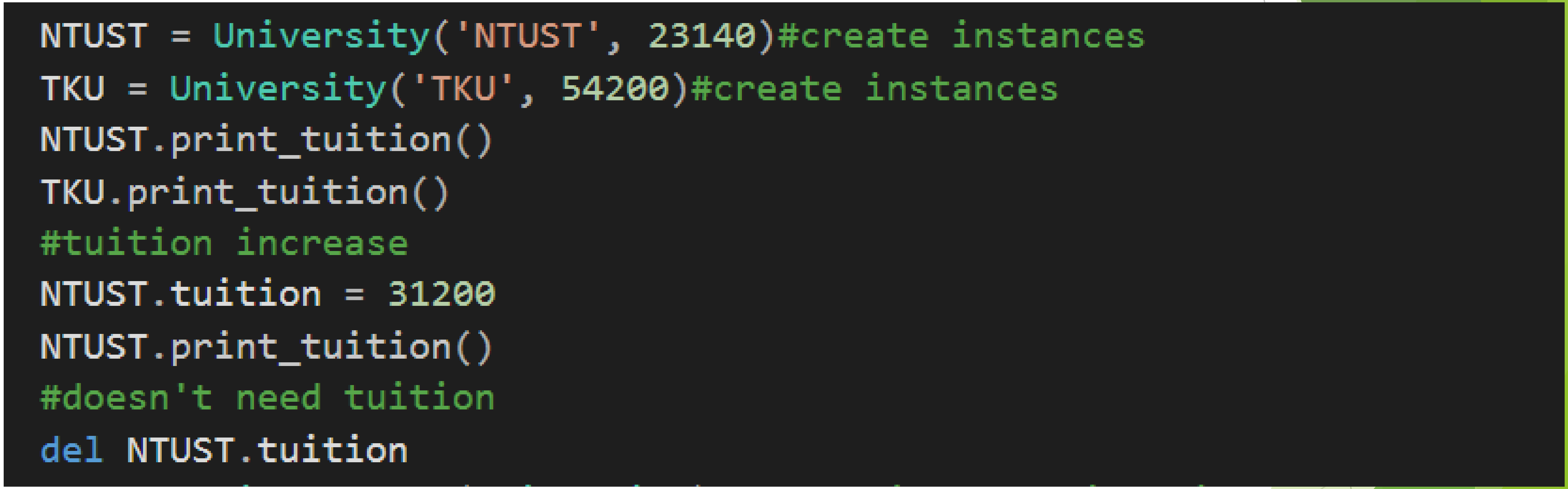
- Instantiation
- Constructor
- Initialization
- Override

Constructor

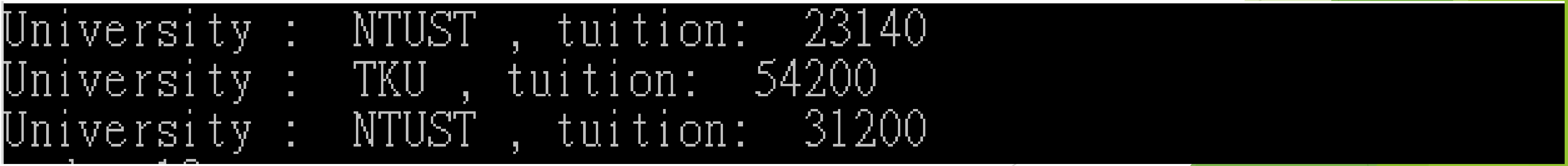


```
class University:
    School_num = 0 #class variable shared in all instances of this class
    __passward = 123 #doesn't visible outside the class
    def __init__(self, name, tuition):#constructor or initialization
        self.name = name
        self.tuition = tuition
        University.School_num += 1   #access class variable

    def print_School_num(self):
        print ("Total School ",University.School_num)

    def print_tuition(self):
        print ("University : ", self.name,  ", tuition: ", self.tuition)
    def __del__(self):
        print ( 'class name : "', self.__class__.__name__, '" destroyed by Garbage Collection' )
```

Instantiation

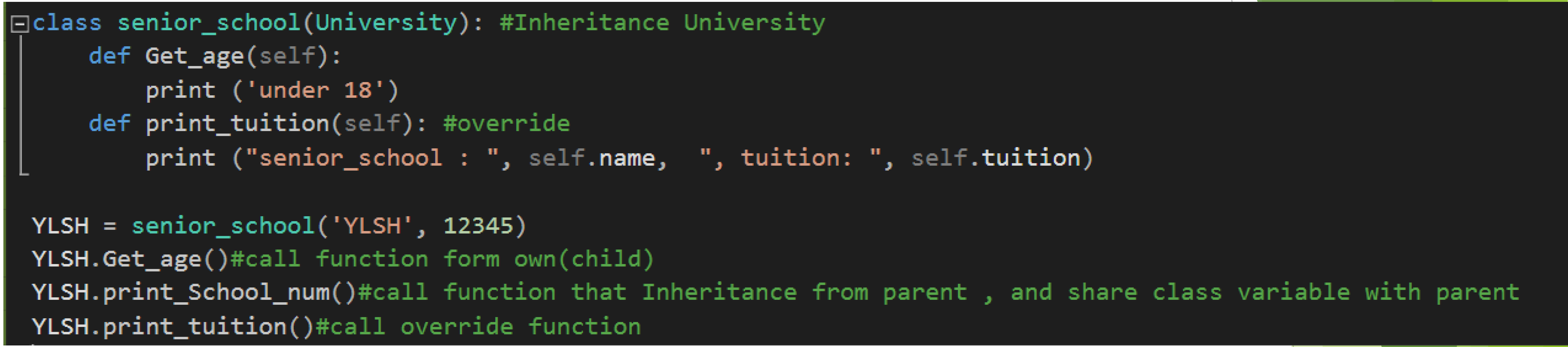


```
NTUST = University('NTUST', 23140)#create instances
TKU = University('TKU', 54200)#create instances
NTUST.print_tuition()
TKU.print_tuition()
#tuition increase
NTUST.tuition = 31200
NTUST.print_tuition()
#doesn't need tuition
del NTUST.tuition
```

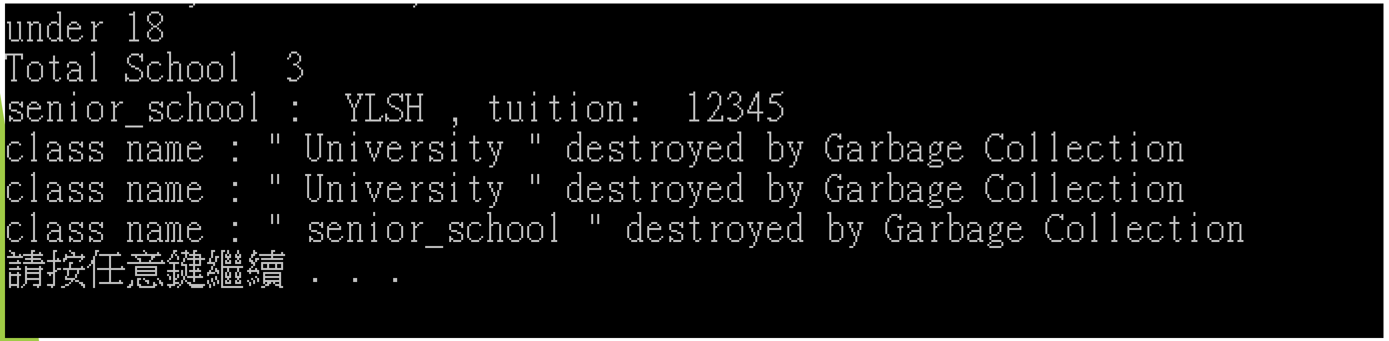


```
University :  NTUST , tuition:  23140
University :  TKU , tuition:  54200
University :  NTUST , tuition:  31200
```

Override and Inheritance



```
class senior_school(University): #Inheritance University
    def Get_age(self):
        print ('under 18')
    def print_tuition(self): #override
        print ("senior_school : ", self.name,  ", tuition: ", self.tuition)

YLSH = senior_school('YLSH', 12345)
YLSH.Get_age()#call function form own(child)
YLSH.print_School_num()#call function that Inheritance from parent , and share class variable with parent
YLSH.print_tuition()#call override function
```

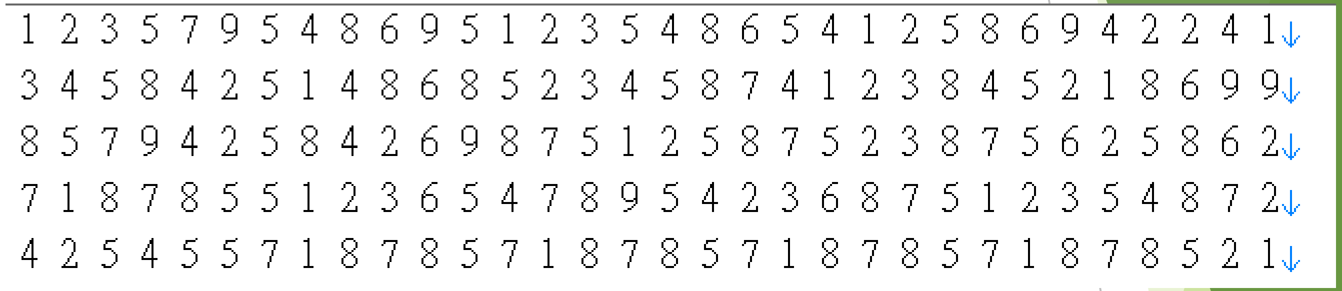


```
under 18
Total School  3
senior_school :  YLSH , tuition:  12345
class name : " University " destroyed by Garbage Collection
class name : " University " destroyed by Garbage Collection
class name : " senior_school " destroyed by Garbage Collection
請按任意鍵繼續 . . .
```

Numpy ndarray

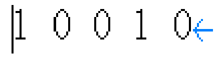
- http://using-python-in-research.site44.com/numpy-mpl-args

Homework



```
1 2 3 5 7 9 5 4 8 6 9 5 1 2 3 5 4 8 6 5 4 1 2 5 8 6 9 4 2 2 4 1
3 4 5 8 4 2 5 1 4 8 6 8 5 2 3 4 5 8 7 4 1 2 3 8 4 5 2 1 8 6 9 9
8 5 7 9 4 2 5 8 4 2 6 9 8 7 5 1 2 5 8 7 5 2 3 8 7 5 6 2 5 8 6 2
7 1 8 7 8 5 5 1 2 3 6 5 4 7 8 9 5 4 2 3 6 8 7 5 1 2 3 5 4 8 7 2
4 2 5 4 5 5 7 1 8 7 8 5 7 1 8 7 8 5 7 1 8 7 8 5 7 1 8 7 8 5 2 1
```

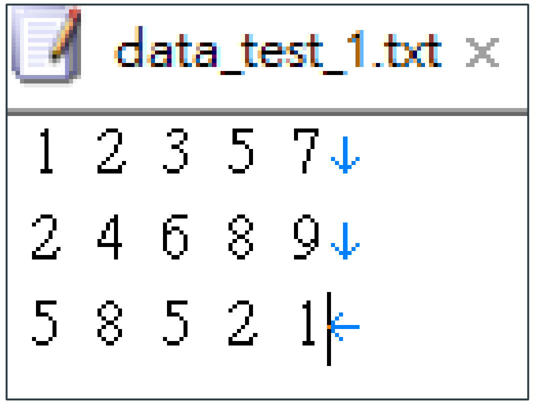
- Broadcasting
- Grade : 2
- Data.txt and Answer.txt
- Read Data.txt and let Data.txt to shape : [5, 32] numpy ndarray
- Read Answer.txt and let Answer.txt to shape : [5, 1] numpy ndarray



```
1 0 0 1 0
```

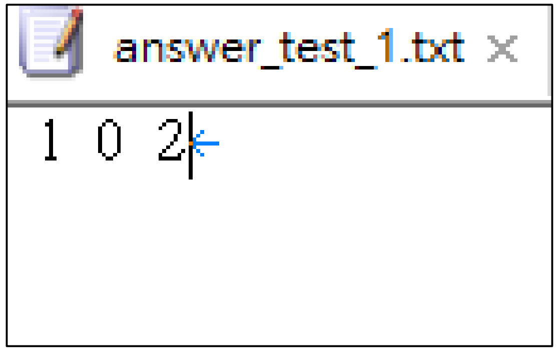
- Add both of them

Test_data and its output(1/2)



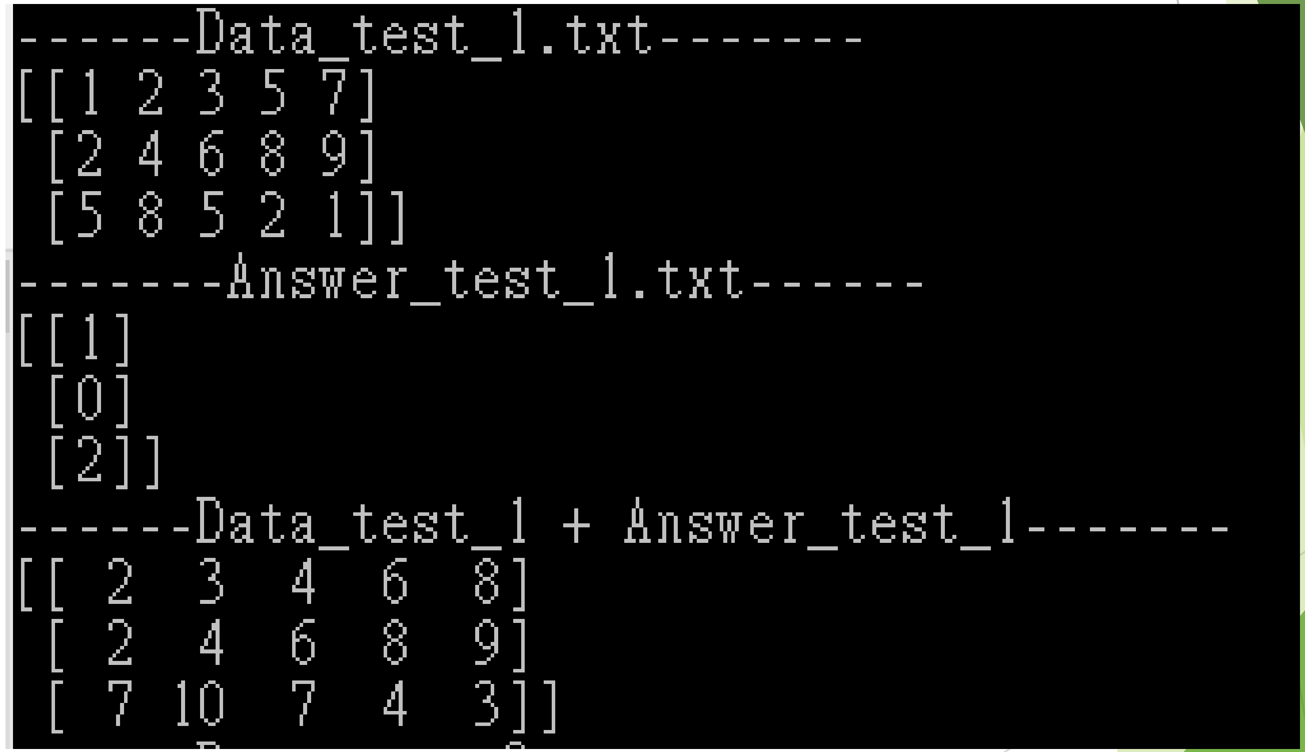
data_test_1.txt

```
1 2 3 5 7
2 4 6 8 9
5 8 5 2 1
```



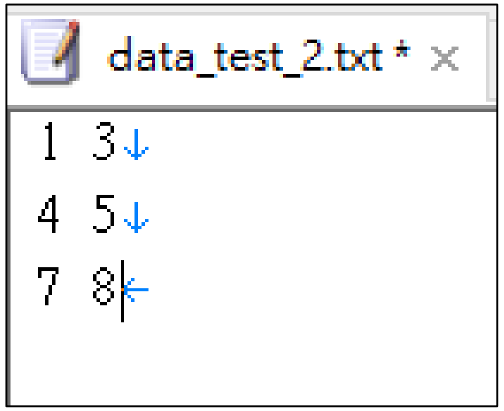
answer_test_1.txt

```
1 0 2
```

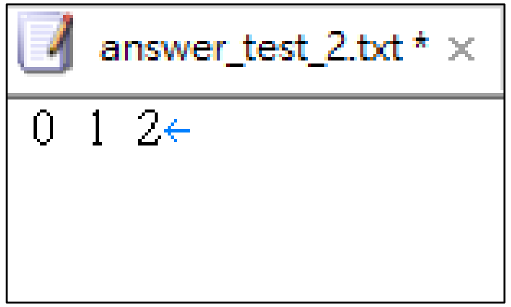


```
------Data_test_1.txt-------
[[1 2 3 5 7]
 [2 4 6 8 9]
 [5 8 5 2 1]]
-------Answer_test_1.txt------
[[1]
 [0]
 [2]]
------Data_test_1 + Answer_test_1-------
[[ 2  3  4  6  8]
 [ 2  4  6  8  9]
 [ 7 10  7  4  3]]
```

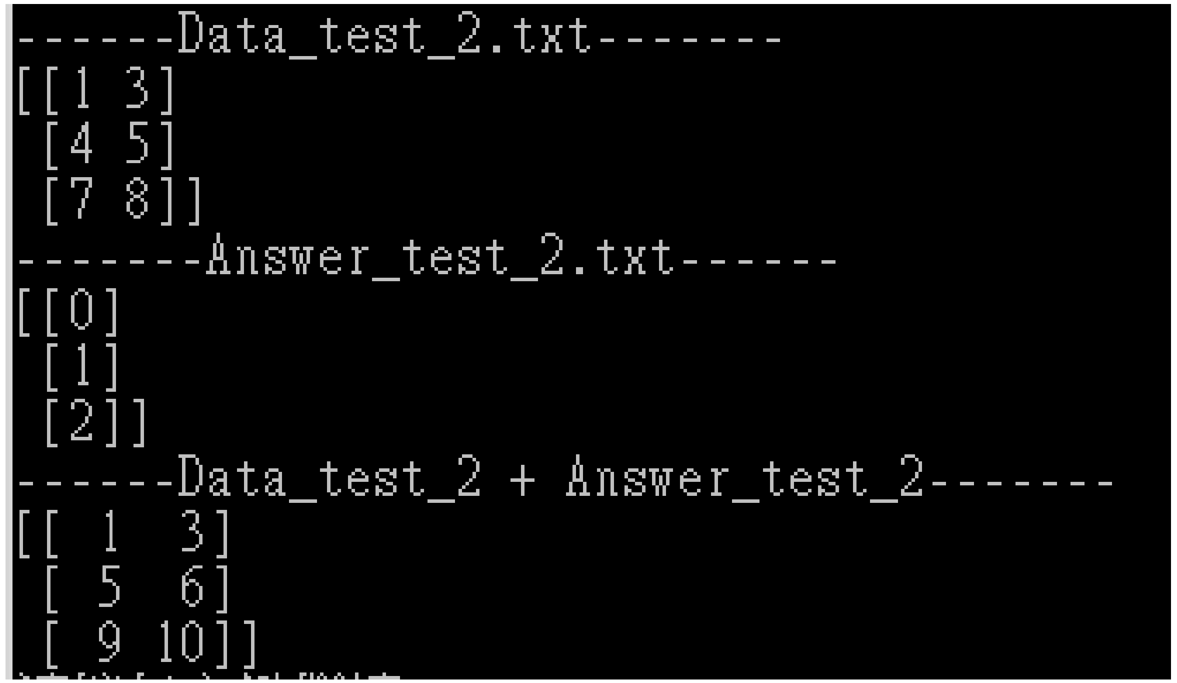
Test_data and its output(2/2)


data_test_2.txt *

```
1 3
4 5
7 8
```


answer_test_2.txt *

```
0 1 2
```



```
------Data_test_2.txt-------
[[1 3]
 [4 5]
 [7 8]]
-------Answer_test_2.txt------
[[0]
 [1]
 [2]]
------Data_test_2 + Answer_test_2-------
[[ 1  3]
 [ 5  6]
 [ 9 10]]
```