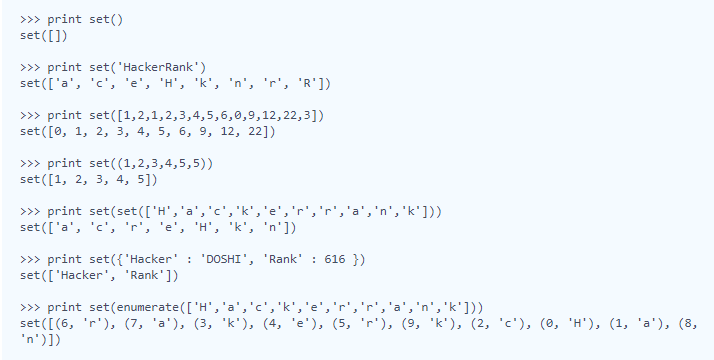
# Sets

## Introduction to Sets

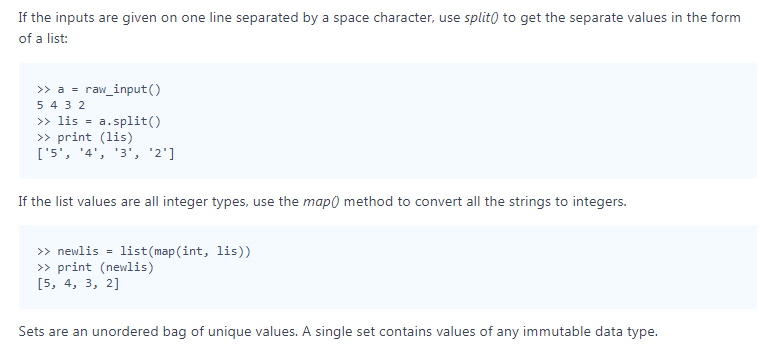


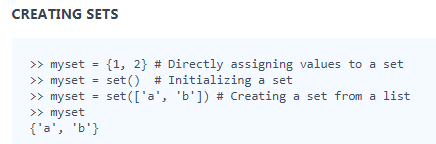
## Symmetric Difference

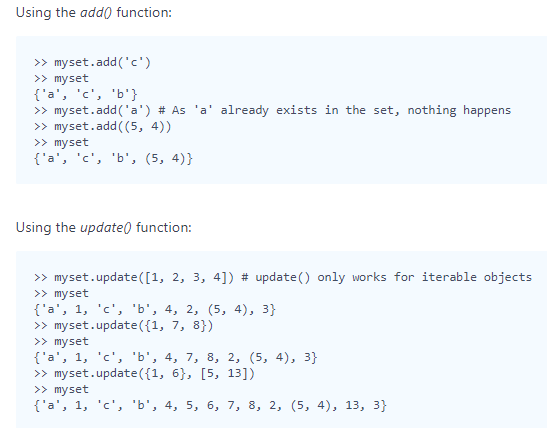
# 缩写for的办法

# set和list的变化教程如上

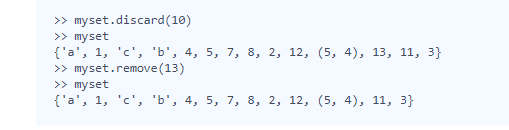
1. **Concept:** Split()， map()



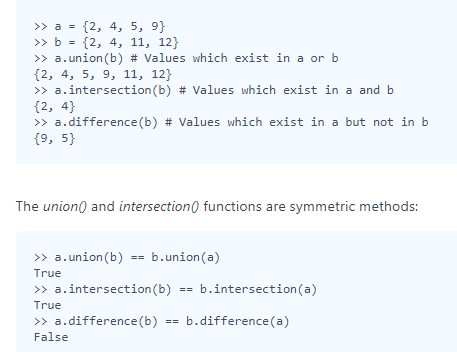
1. 
2. **MODIFYING SETS**: Add(), update()



1. **REMOVING ITEMS** : discard() , remove()



1. **COMMON SET OPERATIONS:** Using union(), intersection() and difference() functions.



## Set .discard(), .remove() & .pop()

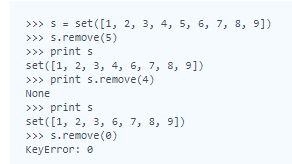
# 注意最后输出的set里元素是数字:9还是字符串:'9'，可以用map函数把字符串转为数字，否则957的和可能变成 "957"而不是21。

# pop的默认删除位置是最后一个，并不随机，所以需要取reverse，否则报错

# reduce（lambda）求和的时候如果list或者set为空集，则报错。sum如果集合为空，结果为0。

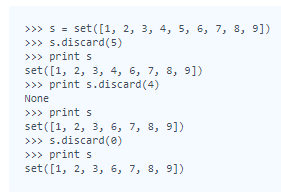
1. **.remove(x)**

This operation removes element x from the set.   
If element x does not exist, it raises a **KeyError**.  
The .remove(x) operation returns **None**



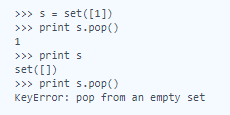
1. **.discard(x)**

This operation also removes element x from the set.   
If element x does not exist, it **does not** raise a KeyError.  
The .discard(x) operation returns **None**.



1. **.pop()**

This operation removes and return an arbitrary element from the set.   
If there are no elements to remove, it raises a **KeyError**.



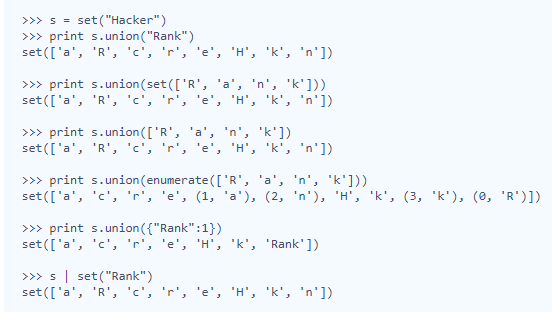
## Set .union() Operation

# enumerate使union的元素标注了位置

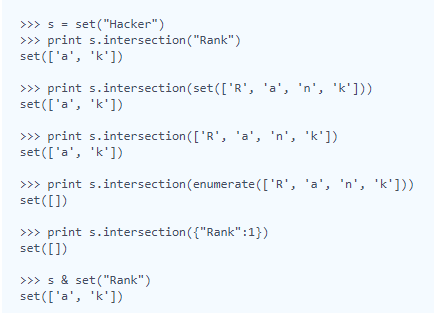
# 注意 {'Rank':1}

# map和raw\_input结合的时候注意split的应用

1. **.union():**The .union() operator returns the union of a set and the set of elements in an iterable. Sometimes, the | operator is used in place of .union() operator, but it operates only on the set of elements in set.

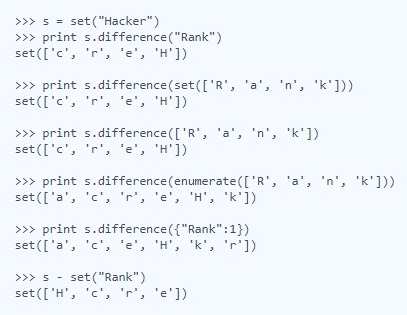
Set is immutable to the .union() operation (or | operation). 

## Set .intersection() Operation

1. **.intersection():**The .intersection() operator returns the intersection of a set and the set of elements in an iterable. Sometimes, the & operator is used in place of the .intersection() operator, but it only operates on the set of elements in set.  
   The set is immutable to the .intersection() operation (or & operation). 

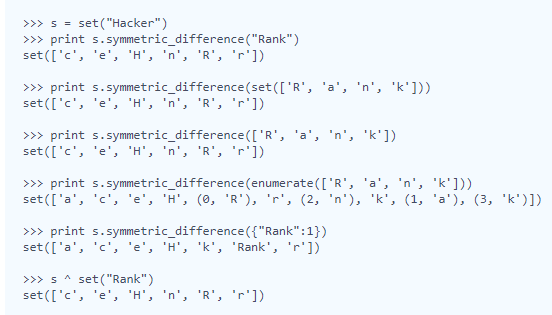
## Set .difference() Operation

1. **.difference():**The tool .difference() returns a set with all the elements from the set that are not in an iterable. Sometimes the - operator is used in place of the .difference() tool, but it only operates on the set of elements in set.  
   Set is immutable to the .difference() operation (or the - operation).



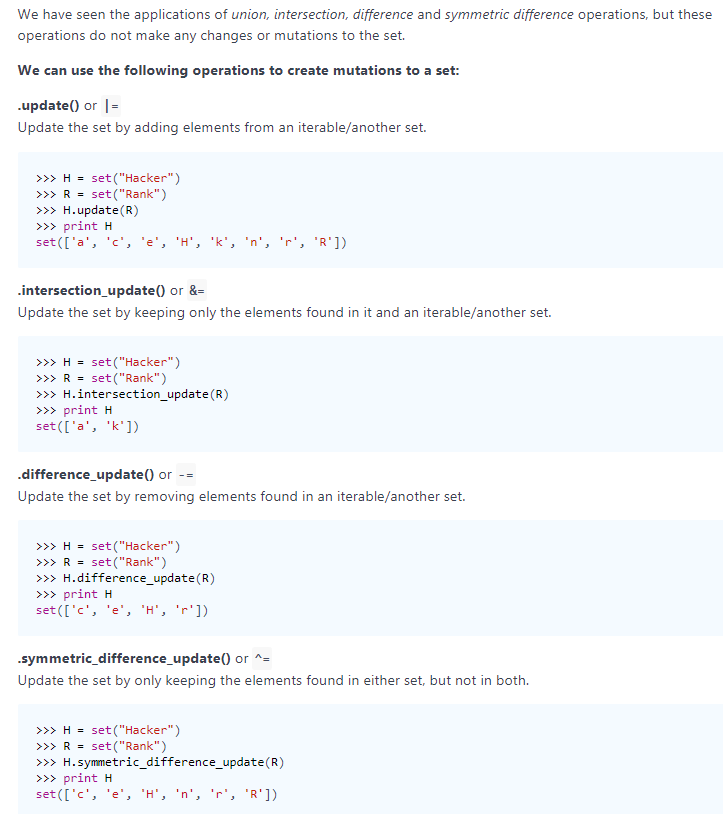
## Set .symmetric\_difference() Operation

1. **.symmetric\_difference():** The .symmetric\_difference() operator returns a set with all the elements that are in the set and the iterable but not both. Sometimes, a ^ operator is used in place of the .symmetric\_difference() tool, but it only operates on the set of elements in set.  
   The set is immutable to the .symmetric\_difference() operation (or ^ operation).



## Set Mutations

# 注意raw\_input()输入的是'9'字符串, map(int, raw\_input())输出的是数字9。



## Check Subset

# 运用issubset函数

## Check Strict Superset

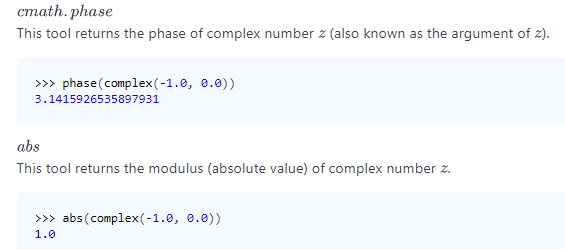
# all for的用法，注意括号的位置

# Math

## Polar Coordinates

# complex(real, imag) 输入极坐标表达式的参数(实部，虚部)

# cmath.polar(number) 将输入的复数表达式转成极坐标形式



## Find Angle MBC

# round：四舍五入；这个函数是内建函数。 round(3.4) # 3.0

#hypot(a,b)：a,b是直角三角形的两个对角边，求斜边长

#atan2 is a function of atan that accepts two inputs

# 直角三角形斜边上的中线等于斜边的一半

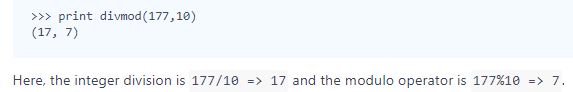
# print str(int(round(math.degrees(math.atan2(AB,BC)))))+'°'：输出弧度然后转化成角度。

## Triangle Quest 2(palindromic triangle)

## Mod Divmod

# from \_\_future import division: from 写在import之前

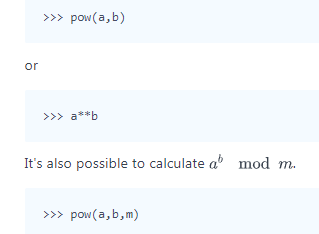
1. Divmod



## Power - Mod Power

# python has a math module that has its own pow(). It takes two arguments and returns a float. Frankly speaking, **we will never use math.pow()**

# pow(a,b,m) a的b次方除以m的余数



## Triangle Quest

# pow()输出int

# Itertools

## itertools.product()

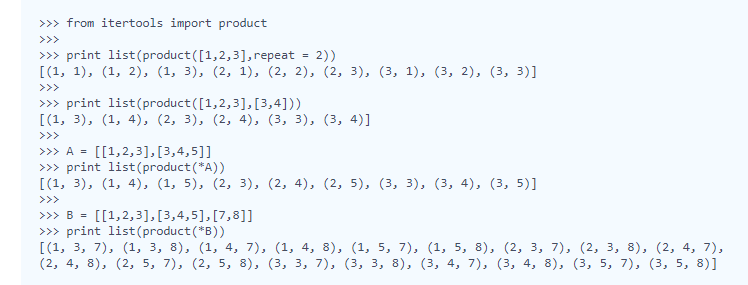
# For example, product(A, B) returns the same as ((x,y) for x in A for y in B).

#一个星号的形式（\*args）用来传递一个可变长度的list参数；两个星号的形式(\*\*args) 用来传递一个可变长度的字典参数

# 在一行上打印list : print(" ".join(str(x) for x in my\_list))

1. [**itertools.product()**](https://www.hackerrank.com/external_redirect?to=https://docs.python.org/2/library/itertools.html#itertools.product)

This tool computes the [cartesian product](https://www.hackerrank.com/external_redirect?to=https://en.wikipedia.org/wiki/Cartesian_product" \t "_blank) of input iterables.   
It is equivalent to nested for-loops.   
For example, product(A, B) returns the same as ((x,y) for x in A for y in B).



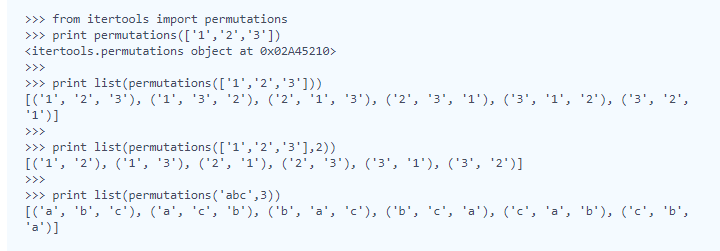
## itertools.permutations()

1. [**itertools.permutations(iterable[, r])**](https://www.hackerrank.com/external_redirect?to=https://docs.python.org/2/library/itertools.html#itertools.permutations)

This tool returns successive r length permutations of elements in an iterable.

If r  is not specified or is None, then r defaults to the length of the iterable, and all possible full length permutations are generated.

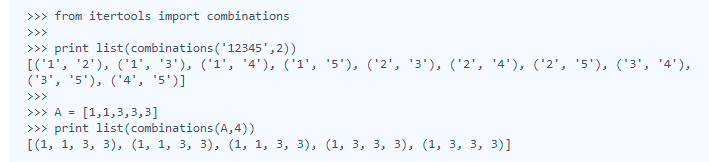
Permutations are printed in a lexicographic sorted order. So, if the input iterable is sorted, the permutation tuples will be produced in a sorted order.



## itertools.combinations()

**1.** [**itertools.combinations(iterable, r)**](https://www.hackerrank.com/external_redirect?to=https://docs.python.org/2/library/itertools.html#itertools.combinations)   
This tool returns the  length subsequences of elements from the input iterable.

Combinations are emitted in lexicographic sorted order. So, if the input iterable is sorted, the combination tuples will be produced in sorted order.



# 跟permutation的不同在于：此处输入字符串，per输入数列

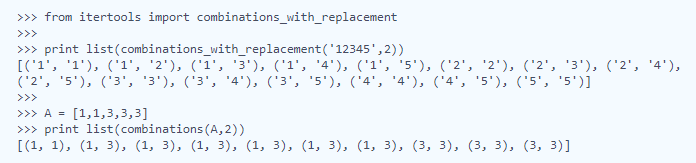
# s , n = raw\_input().split()

## itertools.combinations\_with\_replacement()

# combinations\_with\_replacement: 不止取对应数字后的数字进行组队，也与自己本身组队

1. [**itertools.combinations\_with\_replacement(iterable, r)**](https://www.hackerrank.com/external_redirect?to=https://docs.python.org/2/library/itertools.html#itertools.combinations_with_replacement)   
This tool returns  length subsequences of elements from the input iterable allowing individual elements to be repeated more than once.

Combinations are emitted in lexicographic sorted order. So, if the input iterable is sorted, the combination tuples will be produced in sorted order.



## Compress the String!

# groupby(): You are given a string . Suppose a character 'c' occurs consecutively X times in the string. Replace these consecutive occurrences of the character 'c' with (X,c) in the string.

# [k for k, g in groupby('AAAABBBCCDAABBB')] --> A B C D A B

# [list(g) for k, g in groupby('AAAABBBCCD')] --> AAAA BBB CC D

# zip(): 建立字典>>> x=['bob','tom','kitty']

# >>> y=[80,90,95]

# >>>d=dict(zip(x,y))

# [('bob', 80), ('tom', 90), ('kitty', 95)]

# 如果字典的长度不一致，以短的为准

# dict()： dictary 字典

## Iterables and Iterators

# lambda 是匿名函数 lambda x（函数）: x\*x（函数的内容）

# fliter函数：filter(function, sequence) 。题中的函数是lambda，应用的集合是C

# ''.format：{}是指输出一个字符串，‘：.6’表示保留6位小数 (format 参考网址https://pyformat.info/)

### Itertools包：https://docs.python.org/2/library/itertools.html

###enumerate 用于遍历元素

#list1 = ["这", "是", "一个", "测试"]

#for index, item in enumerate(list1):

# print index, item

#>>>

#0 这

#1 是

#2 一个

#3 测试

###arr1 = [i for i, v in enumerate(arr) if v == 'a' ] 输出数列中特定元素的所有位置

## Maximize It!

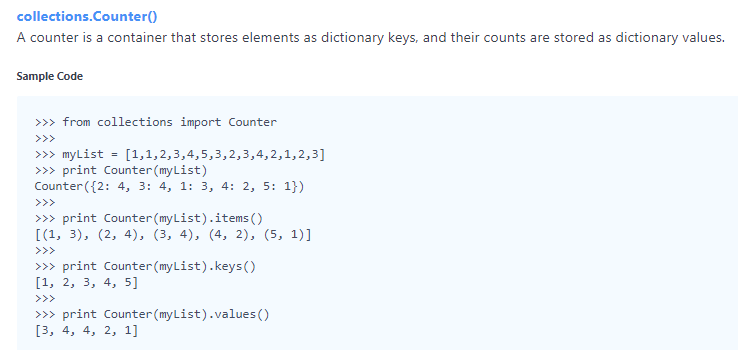
# 使用product函数， conbinations是组合，permutations是排列

# sum(x \*\*2 for x in arr1[k])

# Collections

## collections.Counter()

#Counter： means 计算出现的次数



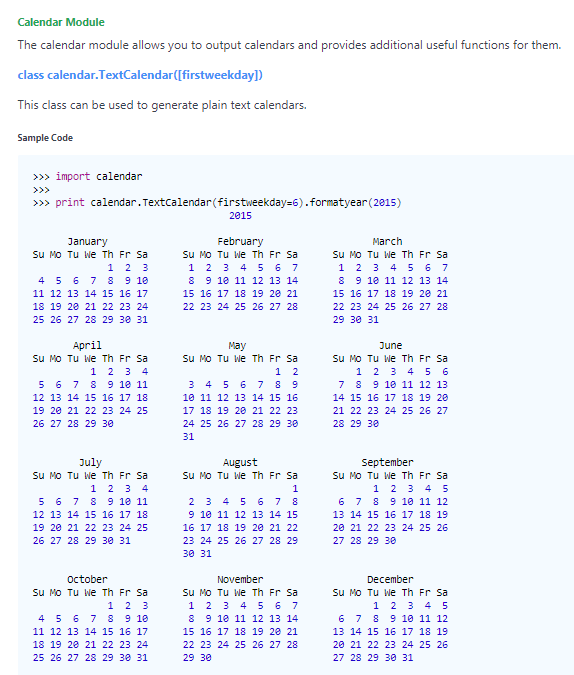
# k,j = map(int,raw\_input().split()) ：使输入值变为整型

# Date and Time

## Calendar Module

# https://docs.python.org/2/library/calendar.html#calendar.setfirstweekday 对于Calendar函数包的使用

# 对字典: '1'是str，1是int。使用不同类型对于的d.values()的排序也不同。



## Time Delta

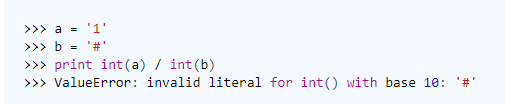
# 时间可以格式化 date\_format = '%a %d %b %Y %X'

# 时间是可以相减的 date1 -= timedelta(hours=int(timezone1[1:3]), minutes=int(timezone1[3:]))

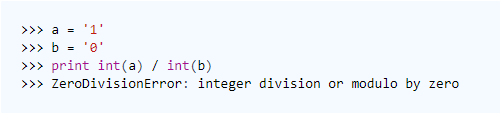
# Errors and Exceptions

## Exceptions

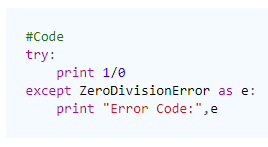
### ValueError: This error is raised when a built-in operation or function receives an argument that has the right type but an inappropriate value.



###ZeroDivisionError: This error is raised when the second argument of a division or modulo operation is zero.



###Handling Exceptions： The statements try and except can be used to handle selected exceptions. A try statement may have more than one except clause to specify handlers for different exceptions.



# Classes

## Classes: Dealing with Complex Numbers

# Complex 为超类，以下的\_\_init\_\_, \_\_add\_\_为子类。

# \_\_init\_\_(self,x,x,x), self是指instance例子本身，x是指instance中存在的变量

# division 的过程: 分子乘上分母，分母乘上分母，实部等于分子实部除以分母实部，虚部等与分子虚部除以分母虚部

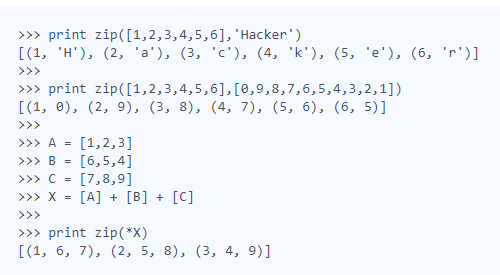
## Class 2 - Find the Torsional Angle

# Should find "cross product" in WIKI,

# Built-Ins

## Zipped!

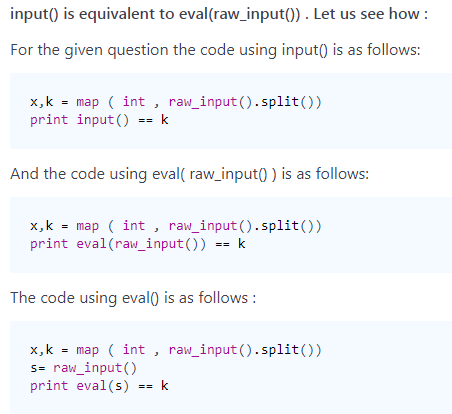
# zip([iterable, ...]) This function returns a list of tuples. If the argument sequences are of unequal lengths, then the returned list is truncated to the length of the shortest argument sequence.



## "\*"在此处使得\*a[i]并不以一个数列输入而是一个个元组

## Input()

# input() = eval(raw\_input()), eval（） 的作用是使括号内的式子生效



# eval() example: http://blog.csdn.net/seetheworld518/article/details/47983511

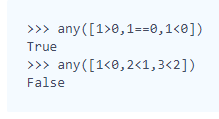
# if "raw\_input()= x\*\*3 + x\*\*2 + x + 1", then "input() = eval('x\*\*3 + x\*\*2 + x + 1')"

## Sort Data

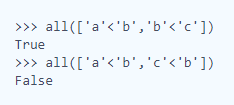
# lambda函数的未知量可以任意取　x,a,b,c,d...

## Any or All

# any(): This expression returns True if **any** element of the iterable is true.   
If the iterable is empty, it will return False.



#all():This expression returns True if **all** of the elements of the iterable are true. If the iterable is empty, it will return True.



# For string type, x== x[::-1] is good at checking "palindromic integer".

## ginortS

### sorted(a, key=lambda x: (x.isdigit(), x.isdigit() and int(x)%2==0, x.isupper(), x.islower(),x)):

# x.isdigit()等判断句，如果成功，把元素放在最后

# x.isdigit()等判断句，放在越前面，优先级越大

# lambda x: ()，此括号内包括所有的判断逻辑, 用","隔开。如若只有一个逻辑，则不用括号。 eg: 'lambda x: x.isdigit()' or 'lambda x:x'

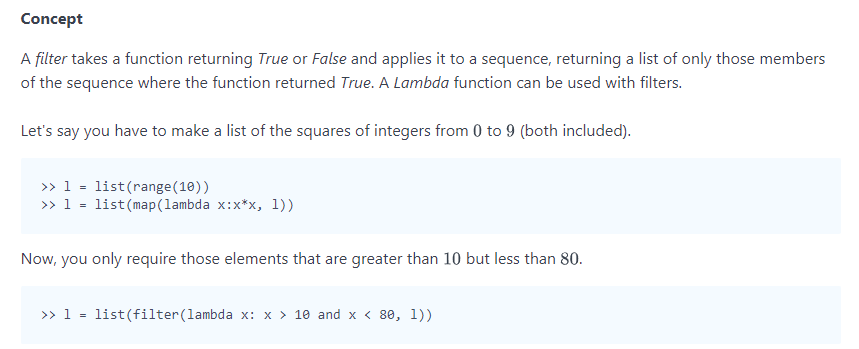
# Python Functionals

## Map and Lambda Function

#yield:相当于return, 此处涉及迭代和生成器. <http://pyzh.readthedocs.io/en/latest/the-python-yield-keyword-explained.html>

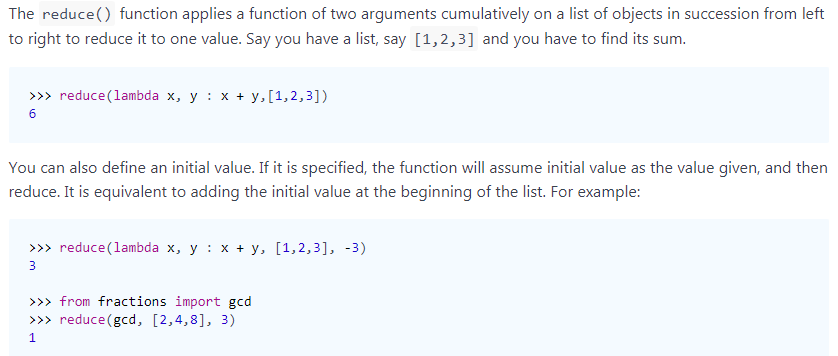
## Validating Email Addresses With a Filter

##A filter takes a function returning True or False and applies it to a sequence, returning a list of only those members of the sequence where the function returned True.



## Reduce Function

# Fraction函数：帮助做约分，eg fractions.Fraction(16, -10) # - 8/5

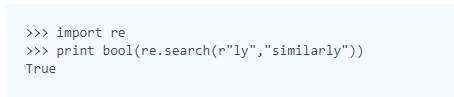
# reduce 可以和lambda一起用，也可以和普通函数一起。eg. reduce(gcd, [2,4,8], 3) 

# Regex and Parsing

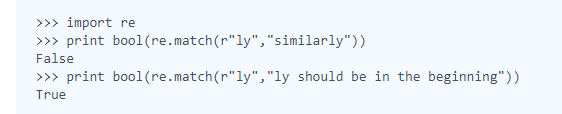
## Introduction to Regex Module

# 对于要使用正则表达式的字符串，全部使用r 前缀，例子http://skypegnu1.blog.51cto.com/8991766/1791680

#re.search(r"ly","similarly")：在第二个字符串中找到第一个，即为true,任意位置。



#re.match(r"ly","similarly")：如果第二个字符串的开头为第一个字符串，则match成功。

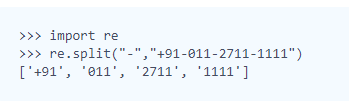


#正则表达式表格：<https://www.cnblogs.com/huxi/archive/2010/07/04/1771073.html>

## Re.split()

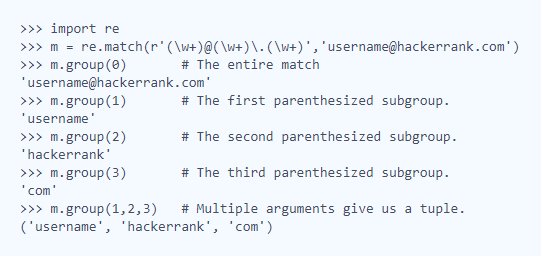
### [re.split()](https://www.hackerrank.com/external_redirect?to=https://docs.python.org/2/library/re.html#re.split)

The re.split() expression splits the string by occurrence of a pattern.

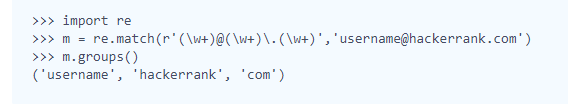


## Group(), Groups() & Groupdict()

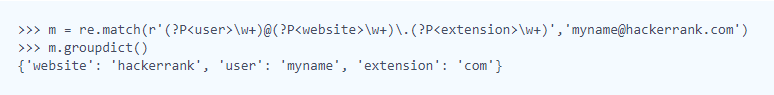
# A group() expression returns one or more subgroups of the match. group(0)为整体输出，group(1)--group(n)为各个部分输出（以圆括号为单位）.



# A groups() expression returns a tuple containing all the subgroups of the match.



# A groupdict() expression returns a dictionary containing all the named subgroups of the match, keyed by the subgroup name.

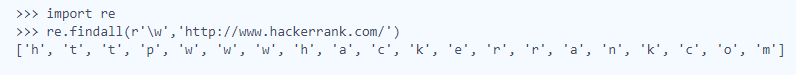


# \1+ matches the same text as most recently matched by the 1st capturing group, eg 组1捕捉到一个2，如果连续出现一个2，则满足\1。 \1+意味着连续至少出现一个2

### isalum()：指字母，包括大小写。

## Re.findall() & Re.finditer()

## re.findall(): The expression re.findall() returns all the non-overlapping matches of patterns in a string as a list of strings.



## re.finditer(): The expression re.finditer() returns an iterator yielding MatchObject instances over all non-overlapping matches for the re pattern in the string.



### 特殊结构 (?=) <https://www.cnblogs.com/huxi/archive/2010/07/04/1771073.html>

### ([AEIOUaeiou]{2,})是正确结构，如若取([AEIOUaeiou]){1,},则会取出包含AEIOUaeiou的子字符串中的最后一个---[a,e,o,o,e]；{2,}类似。

# flags = re.I 表示忽略大小写，所有辅音和元音数组里没有大写字母

###findall中的flag: http://wiki.jikexueyuan.com/project/python-crawler-guide/regular-expressions.html

### if a: 意思是a非空集。

## Re.start() & Re.end()

# re.start(): 是匹配好的第一个数的坐标；re.end()：是匹配字符过后的第一个字符的坐标。

# if not r 如果r为空集

# 如果需要重复search，需要pattern.search函数

## pattern = re.compile(k)

## r = pattern.search(S)

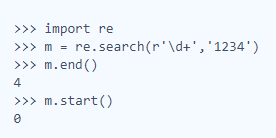
# re.search(pattern, string, flags=0):并没有坐标的选项

# \*pattern.search(string, index\_start, index\_end)

# 不能直接使用k.search, 而需要转化为regex pattern的原因：str没有search函数

### [start() & end()](https://www.hackerrank.com/external_redirect?to=https://docs.python.org/2/library/re.html#re.MatchObject.start)

These expressions return the indices of the start and end of the substring matched by the group.



## Regex Substitution

# 记住(?<=) 和(?=) 是overlapping的搜索，而且不占用位置.

# print 函数等与逐行处理。

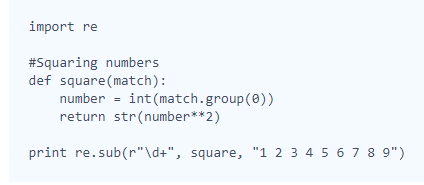
# (?<= ) ：空格就是只打一个空格

# x.group(): 表示匹配pattern的字符串， 对group的解释：http://www.runoob.com/python/python-reg-expressions.html

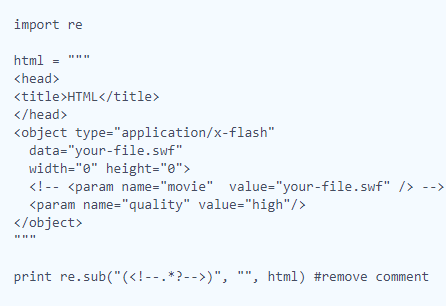
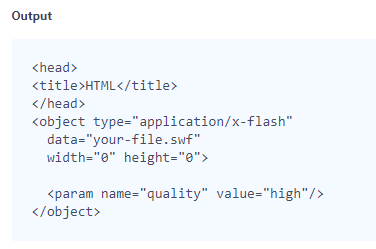
# help(re.search): 当不会使用函数的时候，输入help（fun）会弹出函数的使用方式。

## lambda x: 'and' if x.group()=='&&' else 'or', lambda的简洁表达

1. Re.sub



Output: 1 4 9 16 25 36 49 64 81

**Replacements in Strings**  

## Validating Roman Numerals

# http://www.ttlsa.com/docs/dive-into-python/html/regular\_expressions/n\_m\_syntax.html 罗马数字的正则表达法

## Validating phone numbers

# $表示匹配字符串末尾

## Validating and Parsing Email Addresses

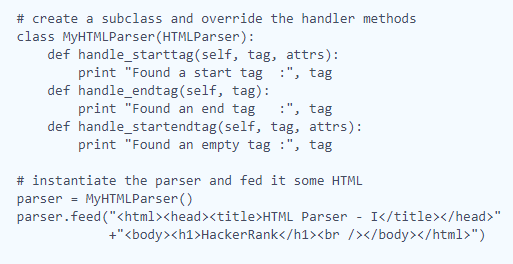
# 注意email 检索的正确地址

# email.utils 函数：

## Hex Color Code

# ()可以避免输出前面的元素

## HTML Parser - Part 1

# start tag: <\*\*\*>; end tag </\*\*\*>; startend tag <\*\* />  

# attrs: 指< >之内除tag以外的元素。

# "if attrs": 如果有attrs, 没有就跳过。

# attrs 里的元素是由编号的。

# html = re.sub(r'(\<\!--.\*?--\>)','',html): for not detect relative info we do not need.

## HTML Parser - Part 2

# if '\n' in comment: 字符串可以作为if的表达输出。

# data.strip()可以去掉\n， if data != '\n' 可以起到同样的效果

## Validating UID

# findall和match的用法

# (.\*[A-Z]){2,}: 表示任意字符加数字的pattern被找到两次，则匹配成功; (.\*[0-9]){3,}类似

# .\*(.).\*\1+.\*: \1+ matches the same text as most recently matched by the 1st capturing group。如果只使用.\*(.).\*\1+，只会匹配到“B1CD1“，而不是B1CD102354

## Validating Credit Card Numbers

# 依然不太理解 (),[],([]) ，元素是元组，字符串，还是以字符串为单位的元组

# (\d)\1{3,}: 用来匹配连续重复元素(注意\1前面一定是())。逻辑是：在数字0到9中找到有重复的数字，随后重复了3次（一共4次）即输出

## Validating Postal Codes

# all((re.match(r'\d{6}$', P), len(re.findall(r"(?<=(\d)\d)\1", P)) <= 1)): all函数的使用

# (?<=(\d)\d)\1：(\d)表示 1st capturing Group, (?<=(\d)\d) 的表达并不吃掉字符，\1表示与1st capturing Group 重复。

# (.)\1：eg.11，检测到的是后面的1，因为检测的是与前面重复的元素。

## Matrix Script

# \r是回车的意思

# [!@#$%& ]+ : 是把找到的连续在一起的满足条件的字符串替换为一个空格

### zip 函数：实际上是分配率，以最短的一个数组的长度为输出长度。范例如下：

>>> a = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

>>> zip(\*a)

>>> [(1, 4, 7), (2, 5, 8), (3, 6, 9)]

###在此：z = [('T', 'h', 'i', 's', '$', '#', 'i')，('s', '%', ' ', 'M', 'a', 't', 'r')，('i', 'x', '#', ' ', ' ', '%', '!')]

# XML

## XML 1 - Find the Score

#node.attrib: 可以找出所有的attributes

## XML2 - Find the Maximum Depth

# depth的意思就是嵌套了几层。

# Closures and Decorators

http://simeonfranklin.com/blog/2012/jul/1/python-decorators-in-12-steps/

1. Function
2. Scope(Name Space)
3. Variable resolution rules (inside function can access global variable, but when we re-assign the global variable inside, we get what we assign default when we print function)
4. Variable lifetime : 函数里的变量在函数输出时消失.
5. Function arguments and parameters : function parameters can be either **positional** parameters that are **mandatory** or **named, default value** parameters that are **optional**. 函数参数的设定**可以给予名字**或不给予；可以给参数**设定默认值（position）**或不设定。
6. Nested functions ：嵌套的函数还是遵循基本规则。
7. Functions are first class objects in Python：函数就是普通的客体，它是可以pass 或者return的。函数也可以被赋值给新的variable，使用需要使用call operator（parentheses）。
8. Closures：函数闭包：外部函数的赋值可以被内部函数引用，不会受lifetime影响。
9. Decorators：应用于bound checking 很方便。
10. The @ symbol applies a decorator to a function: 用于引用bound checking函数。（有特定的格式）
11. \*args and \*\*kwargs： \*表示灵活的位数，\*\*表示dictionaries & key/value pairs （x=1，y=2）
12. More generic decorators：

# Numpy

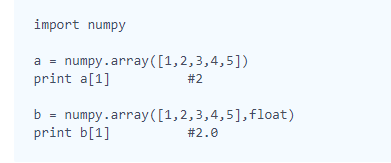
## Arrays

1. # A NumPy array is a grid of values. They are similar to lists, except that every element of an array must be the same type.

# arr[::-1]：倒序。和arr.reverse()的区别：在numpy.array里reverse用不了。

1. [**Arrays**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/arrays.html)

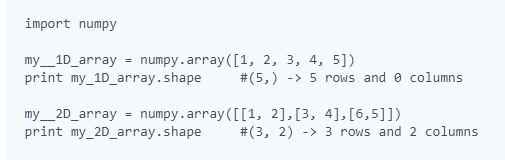
A NumPy array is a grid of values. They are similar to lists, except that every element of an array must be the same type.



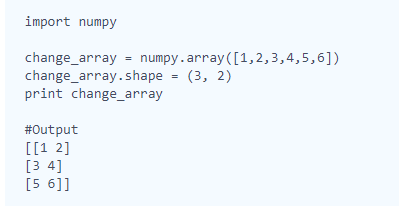
## Shape and Reshape

1. [**shape**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.shape.html#numpy-ndarray-shape)

The shape tool gives a tuple of array dimensions and can be used to change the dimensions of an array.

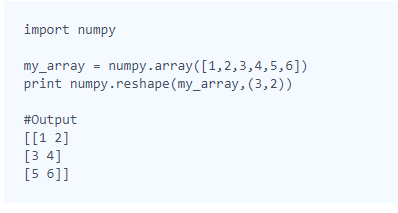
**(a). Using shape to get array dimensions**

**(b). Using shape to change array dimensions**



1. [**reshape**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.reshape.html#numpy.reshape)

The reshape tool gives a new shape to an array without changing its data. It creates a new array and does not modify the original array itself.



## Transpose and Flatten

1. # numpy.array([],[],...) 制造矩阵

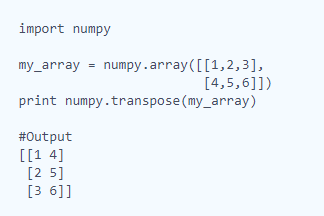
# numpy.transpose(array) 转置矩阵

# array.flatten 可以将矩阵元素按照一元数列打印

##my\_array = numpy.array( [map(int, raw\_input().split()) for i in range(N)] ) 可以代替append 函数

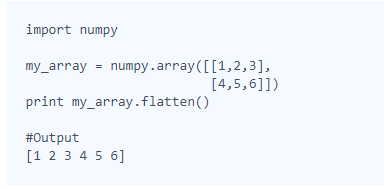
1. [**Transpose**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.transpose.html#numpy-transpose)

We can generate the transposition of an array using the tool numpy.transpose.   
It will not affect the original array, but it will create a new array.



1. [**Flatten**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.flatten.html)

The tool flatten creates a copy of the input array flattened to one dimension.



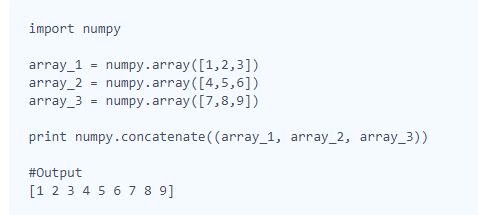
## Concatenate

1. # numpy.concatenate((array1,array2,array3))：可以将array中的元素在一行打印出来。

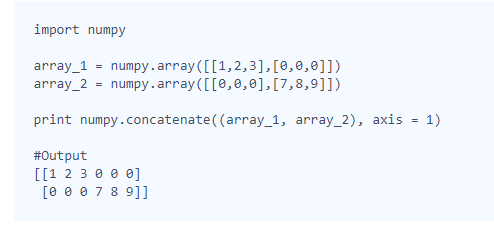
# numpy.concatenate((array\_1, array\_2), axis = 1): 可以将array中的元素打印成两行(每个数列有两个元素 & more than 1 dimension)， 如果改成默认0，则会有4行。

1. [**Concatenate**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.concatenate.html)

Two or more arrays can be concatenated together using the concatenate function with a tuple of the arrays to be joined:



If an array has more than one dimension, it is possible to specify the axis along which multiple arrays are concatenated. By default, it is along the first dimension.



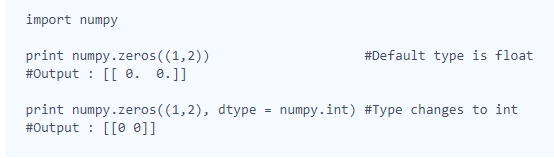
## Zeros and Ones

1. # zeros & ones function

# function 可以用 (x,y,z) ：x表示行，y表示列，z表示重复的次数。

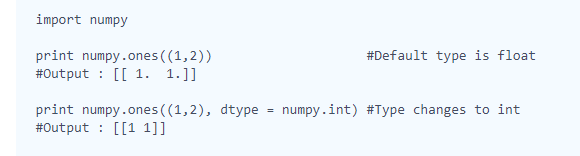
1. [**zeros**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.zeros.html#numpy-zeros)

The zeros tool returns a new array with a given shape and type filled with 0's.



1. [**ones**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.ones.html#numpy-ones)

The ones tool returns a new array with a given shape and type filled with 1's.



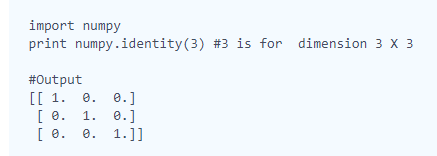
## Eye and Identity

1. # identity(n)：n阶单位矩阵

# eye(x,y,k) ：x\*y dimension, k=1,0,-1表示上对角线，主对角线，次对角线。

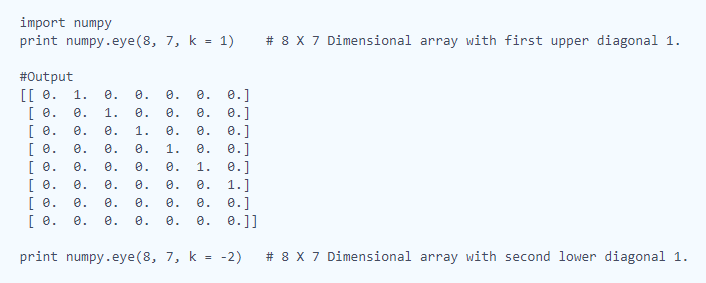
1. [**identity**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.identity.html#numpy.identity)

The identity tool returns an identity array. An identity array is a square matrix with all the main diagonal elements as 1 and the rest as 0. The default type of elements is float.



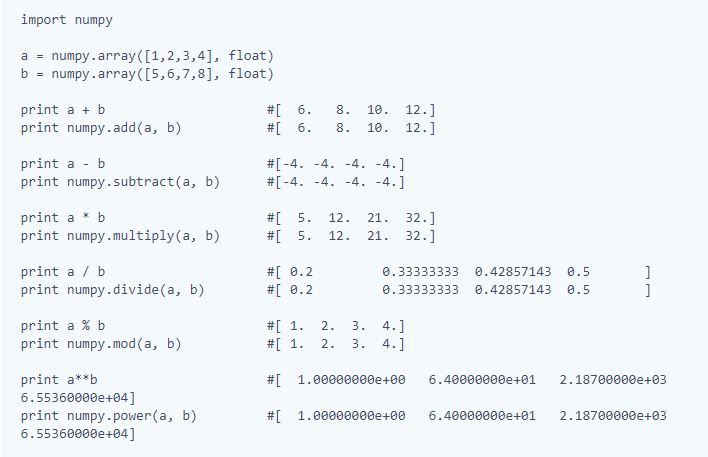
1. [**eye**](https://www.hackerrank.com/external_redirect?to=http://docs.scipy.org/doc/numpy/reference/generated/numpy.eye.html#numpy-eye)

The eye tool returns a 2-D array with 1's as the diagonal and 0's elsewhere. The diagonal can be main, upper or lower depending on the optional parameter k. A positive k is for the upper diagonal, a negative k is for the lower, and a 0 (default) is for the main diagonal.



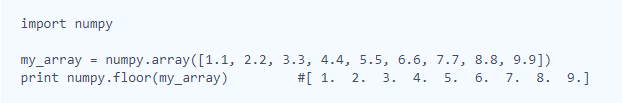
## Array Mathematics

1. # numpy的加减乘除
2. # a = numpy.array([map(int,raw\_input().split()) for i in range(n)]): numpy.array的赋值方法

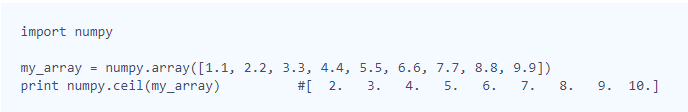


## Floor, Ceil and Rint

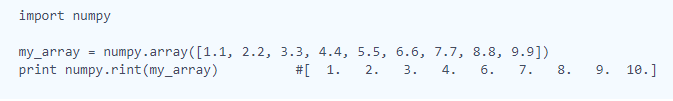
# floor : The floor of x is the largest integer i where i<=x.



# Ceil: The ceiling of x is the smallest integer i where i>=x.

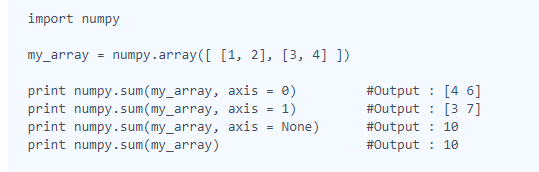


# Rint: rounds to the nearest integer of input element-wise. (四舍五入)

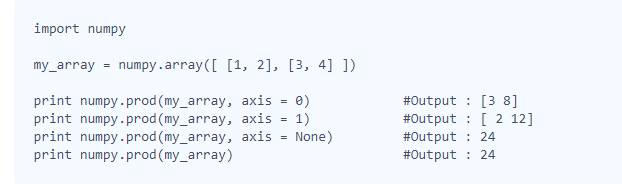


## Sum and Prod

# sum: axis = 0, 列相加， axis = 1， 行相加, axis = None, 所有数字相加。(默认， axis = None)

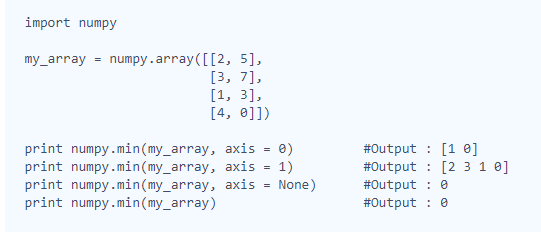


# prod: product， axis跟sum相同。

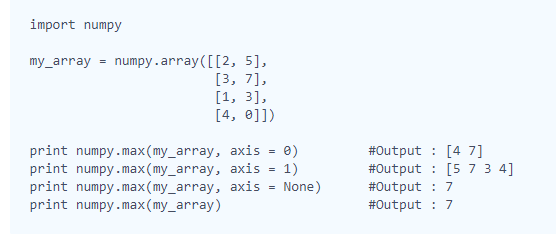


## Min and Max

# min: axis = 0 返回每列的最小值； axis = 1 返回每行的最小值； axis = None 返回矩阵里的最小值。

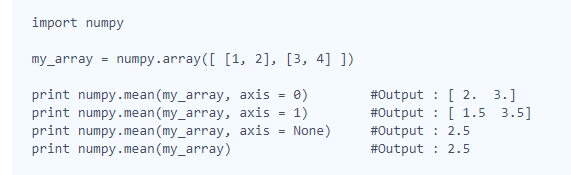


# max: 返回最大值， axis 相同

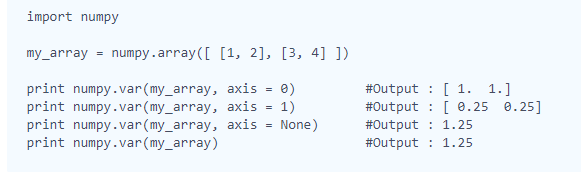


## Mean, Var, and Std

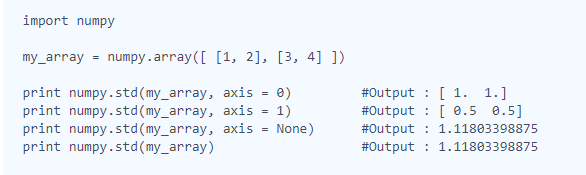
# mean（浮点型）: axis = 0 算出每一列的平均值，axis = 1 算出每一行的平均值， axis = None 算出整个矩阵元素的平均值。默认None。



# var （算数方差）：axis 同上

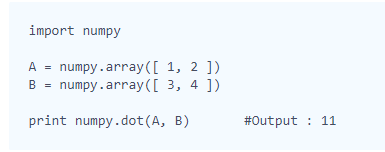


# std （标准差）：axis 同上

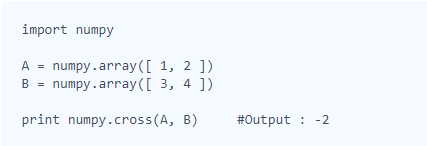


## Dot and Cross

# dot（点乘）

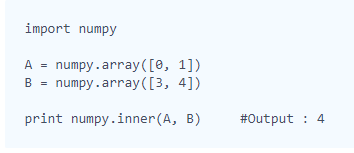


# cross（外积）<https://en.wikipedia.org/wiki/Cross_product>

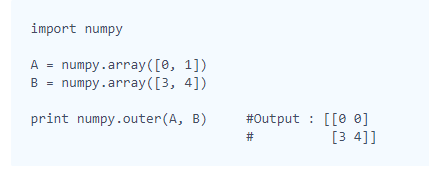


## Inner and Outer

# inner (内积)：对于2D的第一个元素，dot是矩阵1的第一行与矩阵2的第一列乘，inner是矩阵1的第一行与矩阵2的第一行相乘。<https://stackoverflow.com/questions/11033573/difference-between-numpy-dot-and-inner>

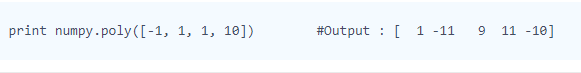


# out（外积）： 对于两个1\*2向量，结果为两个向量每个元素的乘积。

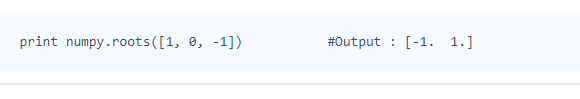


## Polynomials

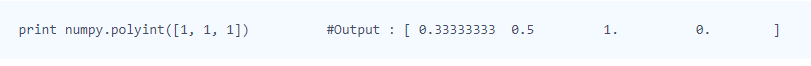
# poly（多项式）: returns the coefficients of a polynomial with the given sequence of roots <https://docs.scipy.org/doc/numpy/reference/generated/numpy.poly.html#numpy.poly>



# roots（多项式）：returns the roots of a polynomial with the given coefficients.



# polyint (反导数，即原函数) : returns an (cofficients of) antiderivative (indefinite integral) of a polynomial given coefficients.



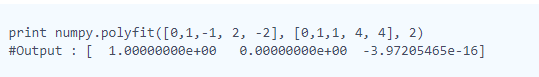
# polyder (导数): returns (cofficients of) the derivative of the specified order of a polynomial.



# polyval: evaluates the polynomial at specific value. <https://docs.scipy.org/doc/numpy/reference/generated/numpy.polyval.html#numpy.polyval>

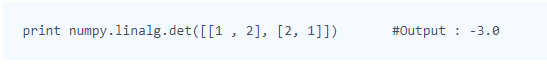


# polyfit: fits a polynomial of a specified order to a set of data using a least-squares approach <https://docs.scipy.org/doc/numpy/reference/generated/numpy.polyfit.html>

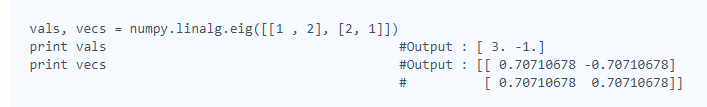


## Linear Algebra

# linalg.det: computes the determinant of an array。只针对2D向量，并且需要两两配对。 <https://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.det.html>



# linalg.eig：computes the (1)eigenvalues and (2)right eigenvectors of a square array。计算特征值和特征向量： http://blog.csdn.net/u010182633/article/details/45921929；特征值是Det(A-lambda\*E)=0的解，可多有重数，有多个；每个lambda带入Av=lambda\*v，可求出一个特征向量。



# linalg.inv: computes the (multiplicative) inverse of a matrix. 利用初等变换，或者A^-1 = A\*/|A|的公式

