[13]:
```
print(13//2)
print(13%2)
print(2 **3)

import math
print(pow(3,3))
```

```
6
1
8
27
```

[11]:
```
num  = 3.145936

print(round(num, 2))
print(f'{num:.2f}')
print(format(num,'.2f'),type(format(num,'.2f')))
print()
print(int(num*100)/100)
```

```
3.15
3.15
3.15 <class 'str'>

3.14
```

[19]:
```
#随机数

import random

random.seed(42) #设定种子使得随机数可以重复

print(random.uniform(3,10)) #随机3~10之间的**小数**
print(random.randint(3,10))#随机3~10之间的**整数**
```

```
7.475987589205186
3
```

[38]:
```
#str

#多行显示

multiLine = """ABC
DEF
GHI"""
print(multiLine)

example = 'I love Python '
print(example.lower())
print(example.upper())
print(example.replace('o', '0'))
print(example.count('o'))
print(example.strip()) #lstrip 或者rstrip 分别去除左边，右边

print(example.isalpha()) #返回是否纯字母 （example有空格）

print('There are', len(example), 'characters in example')
#capitalize （）首字母大写
#title （）所有单词首字母大写
#s[0].upper() + s[1:] 首字母大写其他不动

num = '123'
print(num.isdigit()) #只对字符串起作用

#查找
if 'love' in example:
    print('"love" is in example')
```

```
ABC
DEF
GHI
i love python
I LOVE PYTHON
I l0ve Pyth0n
2
I love Python
False
There are 14 characters in example
True
"love" is in example
```

[39]:
```
# Bool
True and False
```

[39]:
```
# Bool
True and False

#Nonetype
None
```

[41]:
```
# f-string
name = 'Theo'
print(f'{name}, Welcome to Python world')
```

```
Theo, Welcome to Python world
```

[44]:
```
import math

print(math.sqrt(4)) #平方根
```

```
2.0
```

[ ]:
```
from pprint import pprint
pprint()
```

[ ]:
```
#bug 调试
# try/except

# import pdb
# pdb.set_trace()
# 逐行调试
```

[ ]:
```
# While loop
# 条件正确就一直执行，直到条件不符合跳出loop

# pass
# break
```

[ ]:
```
#List 列表序号从【0】开始
lst1.extend(lst2)
lst1.append(lst2)
lst3 = lst1.copy() #复制lst1 且后续对lst3的操作不影响lst1
lst3 = lst1 #lst1和lst3对应同一个list
lst1.remove(element)

lst1.pop() #弹出且默认弹出最后一个列表元素
           #输入数字，按照index的顺序弹出相应位置 lst1.pop(1)弹出第二个元素

lst1.sort()#排序，默认以升序排列 降序 reverse =True，保留原列表（不对原来列表有影响）使用sorted(); 注意sort()本身返回None值
           #排序对大小写敏感，先大写再小写 sort(key = str.lower)不受到大小写影响
           # key = len 根据字符长度排序
           #key = abs 按照绝对值排序
           # key=lambda x: (...) 中可以写多个条件
           # 想降序某个条件就用负号 -x['score']，或者配合 reverse=True
           # 字符串排序默认区分大小写（大写排在前面），可以用 .lower() 做统一处理
del lst1
lst1.clear()
lst1.count(element)
lst1.reverse()

in,not in

range(start, end, step) #生成一个从0到i（不包括i）的List


# 获取列表索引
lst1.index(element)

# enumerate() 是 Python 的一个内置函数，用于在遍历可迭代对象（比如列表、元组、字符串等）时，同时获取元素的索引和值，非常方便。
# 例子
fruits = ['apple', 'banana', 'cherry']

for index, fruit in enumerate(fruits):
    print(index, fruit)

# 特定位置插入元素
lst1.insert(index, element)
```

[43]:
```
# For loop
# 条件错误就反复执行,正确跳出loop

break
continue
pass
```

```python
# For loop
# 条件错误就反复执行,正确跳出loop

break
continue
pass
```

```python
# Dict 字典

dict1 = {key:value}
dict2 = dict(key = value) #key在dict中是唯一的，重复的key，后一个会覆盖前一个key

# 添加元素
dict1[key] = value

# change value
dict1.update({key: value2})

dict1.pop(key)
dict1.clear(key)
del dict2

len(dict1)

dict1.items()
dict1.keys()
dict1.values()

# 子字典

# 将两个列表糅合成一个字典
key = []
value = []

dict2 = dict(zip(key,value))

# 需要列表一样索引key，value时候.\比如足球比赛那道例题

list(dict1.items()) #获取字典每一项，为元组 tuple
list(dict1.keys()) #获取字典key成list
list(dict1.values())#获取字典value成list
```

```python
# 自定义函数

def function_name(param1, param2 = 'example', .....):
    global param1
    pass #return None
    return result

def name(*args):#Arbitrary Arguments：把所有的输入值聚成一个tuple。不可改变的list，支持输入多个值
    print(args)
    print(type(args))

name("David", "John", "Smith", "Denise")

def construct_login_greeting(**kargs):#Arbitrary Keyword Arguments: 支持输入内容定义，输出为一个字典
    print(kargs)
    print(type(kargs))

construct_login_greeting(fname = "David", m1name = "John", m2name= "Smith", lname = "Denise")

#匿名函数 临时函数 Anonymous Functions

print((lambda num: num*num*num)(9)) #def calculate_cube(num):
                                    # return num*num*num
                                    # calculate_cube(9)

lambda_cube = lambda num: num*num*num#等效
print(lambda_cube(9))
```

```python
# Str slices
```

```
# Str slices
str1 = 'HELLO WORLD!'
# str[start: stop: step]
# str1[ : :-1]-> !DLROW OLLEH
# str1[-3: ]-> LD!
# str1[ :5]-> HELLO

str.split(separator, maxsplit) #以什么为分割，分割多少次 分割2次，为三份
                               #输出List

'#'.join(lst1) #将List合并，以'#'（任何符号或者空符号或者空格均可以）来分割list中的元素
```

```
# Tuples  immutable
tuple = ()

-------------------------------------------
#Sets unindexed & unordered & unchangeable but can remove or add item
set = {}

# set添加元素用add()
set.add()
#并集:合并并返回所有去重复后的元素
a.union(b) #等效 a|b

# 交集：返回两个都包含的元素
a.intersection(b) #等效 a&b

# 返回只有集合1包含但集合2不包含的元素
a.difference(b) #等效 a-b

#a是否是b的子集
a.issubset(b) #等效 a <=b

#a 是否为b的超集/父集
a.issuperset(b) #等效 a>=b

#检查是否没有交集
a.isdisjoint(b) #没有返回True,有则返回False
```

```
data = input("请输入列表元素，用逗号分隔：") # 如：1,2,3,4
lst = data.split(",") # 变成 ['1', '2', '3', '4']
lst = [int(x) for x in lst] # 转换为整数列表 [1, 2, 3, 4]
print(lst)


data = input("请输入字典项（key:value 用逗号分隔）：") # 如：a:1,b:2
items = data.split(",") # ['a:1', 'b:2']
d = {k: int(v) for k, v in (item.split(":") for item in items)}
print(d)
```