**PriorityQueue**

from queue import PriorityQueue

PriorityQueue使用heapq同样的实现方式，具有相同的复杂度。但在其基础上，支持异步调用，所以支持多线程。

成员函数：

get():取队首元素的值并弹出，注意优先队列为空时调用get()不会报错而是等待到取出元素。

put():将元素放入优先队列，可以放入元组，以第一个值为排序关键字。

full():判断队列是否满

empty():判断队列是否空

也可以将自定义元素加入队列中，需要在class中重载\_\_lt\_\_(self, other)函数。

例如

class node:

def \_\_init\_\_(self, val, x):

self.val = val

self.x = x

def \_\_lt\_\_(self, other):

return self.val < other.val

**heapq**

import heapq

堆的概念：具有特定性质的二叉树，分为大根堆和小根堆，大根堆满足性质为根节点的值大于子节点的值，小根堆同理。

1. heappush(heap, item)建立大、小根堆

向heap中添加item，默认建立小根堆，建立大根堆可以使用给元素加负号的做法，使得最小值变成最大值，在使用时取负。

1. heapify(heap)转换大小根堆

在线性时间内将heap列表转化为小根堆

1. heappop(heap)

从heap中弹出并返回最小元素。可以根据这个方法解决堆排序。普通list调用此方法仅仅返回第一个值。

1. heappushpop(heap, item)同时push和pop

先将item插入到heap中并保持堆性质，然后弹出最小元素。

1. heapreplace(heap, item)

与heappushpop相反，向进行heappop，再进行heappush，注意heap为空时会报错

1. merge(\*iterables)多个合并

合并多个堆然后输出，如果输入是无序list，则merge后也无序。

1. nlargest(n, iteabale, key=None)和nsmallest(n, iterbale, key=None)

获取列表中最大最小的几个值，key的作用类似于sorted中的key。例如字典选择val，或者元组选择第2个值。

**defaultdict**

from collections import defaultdict

使用defaultdict可以得到初始化数据类型的字典，例如int, list, dict等

defaultdict(int)得到默认val=0的字典，list则默认val=[], dict则默认val={}。

**Sort**

python自定义的排序有两种方式：x.sort()进行原地排序，即排序后的结果保存在x中，以及sorted(x)对x进行排序后返回有序的列表。

而如果想自定义排序函数的话，类似于C++等语言，需要传入一个自定义的排序函数cmp(x, y)，在其中定义规则并以cmp=的参数传入排序函数。但在py3中，cmp关键字被移除了，需要使用**functools.cmp\_to\_key**(cmp)传递给key参数完成，例如a.sort(key= **functools.cmp\_to\_key**(cmp))，或者通过lambda定义关键字，如sort(key=lambda x: (x[2], x[1]))表示先按照x[2]排序，如果相等则按照x[1]排序。

或者也可以使用operator指定对象，如itemgetter指定下标，attrgetter指定属性名。