How to put items into priority queues?

Asked 8 years, 7 months ago Active 1 year, 5 months ago Viewed 53k times

34

13

In the Python docs,

> The lowest valued entries are retrieved first (the lowest valued entry is the one returned by `sorted(list(entries))[0]`). A typical pattern for entries is a tuple in the form: `(priority_number, data)`.

It appears the queue will be sorted by priority then data, which may not be always correct. Suppose data "item 2", is enqueued before "item 1", item 1 will still go first. In another docs page, heapq, it suggests the use of a counter. So I will store my data like `entry = [priority, count, task]`. Isn't there something like

```
PriorityQueue.put(item, priority)
```

Then I won't need to implement the ordering myself?

python queue Edit tags

edited Feb 15 '12 at 8:18
agf
140k 32 260 222

asked Feb 15 '12 at 7:47
Jiew Meng
69.9k 157 428 724

3 Answers

Active Oldest Votes

29

As far as I know, what you're looking for isn't available out of the box. Anyway, note that it wouldn't be hard to implement:

```
from Queue import PriorityQueue

class MyPriorityQueue(PriorityQueue):
    def __init__(self):
        PriorityQueue.__init__(self)
        self.counter = 0

    def put(self, item, priority):
        PriorityQueue.put(self, (priority, self.counter, item))
        self.counter += 1

    def get(self, *args, **kwargs):
        _, _, item = PriorityQueue.get(self, *args, **kwargs)
        return item


queue = MyPriorityQueue()
queue.put('item2', 1)
queue.put('item1', 1)

print queue.get()
print queue.get()
```

Example output:

```
item2
item1
```

answered Feb 15 '12 at 8:01
jcollado
34.1k 5 89 128

I am new to Python, to me, `PriorityQueue.__init__(self)` appears to be calling a static method. Is there something like `parent.__init__(self)`? – Jiew Meng Feb 15 '12 at 11:09

Oh, or should I interpret it as: I am calling the static method passing in `self`. If I call the object method, I won't need that? – Jiew Meng Feb 15 '12 at 11:11

1 `PriorityQueue.__init__(self)` is more properly written `super(MyPriorityQueue, self).__init__()`, but it is valid. – Fred Foo Feb 15 '12 at 11:32

3 @larsmans I tried that, but it seems that `PriorityQueue` is an old style class and it doesn't work with `super`. – jcollado Feb 15 '12 at 12:06

It appears you're right. Rather surprising; please excuse my previous comment. – Fred Foo Feb 15 '12 at 12:26

|

1

I made something like this to accomplish a FIFO similar to gfortune, but without the need of calling time.time() everywhere: (Python 3 only)

```
import time
from dataclasses import dataclass, field

@dataclass(order=True)
class PrioritizedItem:
    prio: int
    timestamp: float = field(init=False, default_factory=time.time)
    data: object = field(compare=False)
```

Now you can do:

```
import queue

item1 = PrioritizedItem(0, "hello world")
item2 = PrioritizedItem(0, "what ever")
q = queue.PriorityQueue()
q.put(item1)
q.put(item2)
```

And be sure, they will always be extracted in the same order.

edited Apr 13 '19 at 17:04
marc_s
650k 146 1226 1355

answered Apr 9 '19 at 12:24
Rene
11 2

34

Just use the second item of the tuple as a secondary priority if a alphanumeric sort on your string data isn't appropriate. A date/time priority would give you a priority queue that falls back to a FIFIO queue when you have multiple items with the same priority. Here's some example code with just a secondary numeric priority. Using a datetime value in the second position is a pretty trivial change, but feel free to poke me in comments if you're not able to get it working.

Code

```
import Queue as queue

prio_queue = queue.PriorityQueue()
prio_queue.put((2, 8, 'super blah'))
prio_queue.put((1, 4, 'Some thing'))
prio_queue.put((1, 3, 'This thing would come after Some Thing if we sorted by this text
entry'))
prio_queue.put((5, 1, 'blah'))

while not prio_queue.empty():
    item = prio_queue.get()
    print('%s.%s - %s' % item)
```

Output

```
1.3 - This thing would come after Some Thing if we didn't add a secondary priority
1.4 - Some thing
2.8 - super blah
5.1 - blah
```

Edit

Here's what it looks like if you use a timestamp to fake FIFO as a secondary priority using a date. I say fake because it's only approximately FIFO as entries that are added very close in time to one another may not come out exactly FIFO. I added a short sleep so this simple example works out in a reasonable way. Hopefully this helps as another example of how you might get the ordering you're after.

```
import Queue as queue
import time

prio_queue = queue.PriorityQueue()
prio_queue.put((2, time.time(), 'super blah'))
time.sleep(0.1)
prio_queue.put((1, time.time(), 'This thing would come after Some Thing if we sorted by
this text entry'))
time.sleep(0.1)
prio_queue.put((1, time.time(), 'Some thing'))
time.sleep(0.1)
prio_queue.put((5, time.time(), 'blah'))

while not prio_queue.empty():
    item = prio_queue.get()
    print('%s.%s - %s' % item)
```

edited Feb 15 '12 at 8:31

answered Feb 15 '12 at 8:12

gfortune
2,339 11 12