Mod 10 Lab: Priority Queues

Implement the Priority Queue ADT with two (non-heap) data structures:

- `PQ_UL` - unordered list
- `PQ_OL` - ordered list

For what it's worth, you can find partial or full solutions for this lab in the textbook and video lectures for this module. The problems were chosen for lab because they are good learning exercises.

To optimize your learning, avoid external resources during lab and figure things out with yourself, your partner, and your TA. Feel free to access other resources after lab if you do not finish during. ## Entry

Your Priority Queues can both use the same Entry class.

Magic Methods

- `init` - gives entries (at least) two attributes:
  - `item`
  - `priority`
- `lt` - returns `True` if `self` has a higher priority (a lower number) than `other`, `False` otherwise
- `eq` - returns `True` if two entries have the same priority *and* item

Priority Queue

Each priority queue you implement should support the following ADT

Magic Methods

- `len`
  - returns the number of entries in the priority queue

Non-magic Methods

- `insert(item, priority)`
  - adds `item` with given `priority` to priority queue
- `find_min()`
  - returns (but does not remove) the `item` with minimum `priority`
- `remove_min()`
  - returns and removes the `item` with minimum `priority`

Special Cases/Notes

- Feel free to use the `lst.sort()` method in this assignment.

Examples

Any examples below are intended to be illustrative, not exhaustive. Your code may have bugs even if it behaves as below. Write your own tests, and think carefully about edge cases.

Unordered list - sequential insertion

```
>>> from lab10 import *
>>> pq = PQ_UL()
>>> n = 100
>>> l = [i for i in range(n)]
```

```
>>> for i in range(n):
...     pq.insert(str(i), i)
...
>>> print(len(pq))
100
>>> old = pq.remove_min()
>>> for i in range(1, n):
...     peek = pq.find_min()
...     new = pq.remove_min()
...     assert new == peek
...     assert old.priority <= new.priority
...     old = new
...
>>> print(len(pq))
0
```

Ordered list - random insertion

```
>>> from lab10 import *
>>> import random
>>> pq = PQ_OL()
>>> n = 100
>>> l = [i for i in range(n)]
>>> random.shuffle(l)
>>> for i in range(n):
...     pq.insert(str(l[i]), l[i])
...
>>> print(len(pq))
100
>>> old = pq.remove_min()
>>> for i in range(1, n):
...     peek = pq.find_min()
...     new = pq.remove_min()
...     assert new == peek
...     assert old.priority <= new.priority
...     old = new
...
>>> print(len(pq))
0
```

Submitting

At a minimum, submit the following files:

- `lab10.py`
  - contains classes
    - `PQ_UL` - unordered list
    - `PQ_OL` - ordered list

Students must submit to Mimir **individually** by the due date (typically, Sunday at 11:59 pm EST) to receive credit.

Grading

- 50 - `PQ_UL`

  - 25 - sequential insertion
  - 25 - random insertion
- 50- `PQ_OL`
  - 25 - sequential insertion
  - 25 - random insertion

Feedback

If you have any feedback on this assignment, please leave it here.

We check this feedback regularly, and it has resulted in many improvements.