

## Priority queue

### 1.insert

```
from queue import PriorityQueue

pq = PriorityQueue()

pq.put((2, "Harry"))
pq.put((3, "Charles"))
pq.put((1, "Riya"))
pq.put((4, "Stacy"))

while not pq.empty():
    print(pq.get())
```

### 2.remove

```
from queue import PriorityQueue

# Create PriorityQueue
pq = PriorityQueue()
pq.put((2, "Harry"))
pq.put((3, "Charles"))
pq.put((1, "Riya"))
pq.put((4, "Stacy"))

# Item to remove
item_to_remove = (3, "Charles")

remaining_items = [] # store items that are NOT removed

# Extract items and filter
while not pq.empty():
    item = pq.get()
    if item != item_to_remove:
        remaining_items.append(item)

# Display only items not removed
print("Remaining elements:", remaining_items)
```

### 3.from queue import PriorityQueue

```
# Simulated customer query
customer_query = "I want to return my headphones"

# Candidate responses with scores
# Format: (urgency_score, purchase_match_score, browsing_match_score, response_text)
responses = [
    (1, 0.9, 0.2, "You can start a return for your headphones here."),
    (3, 0.8, 0.5, "Looking for new headphones? Check these out."),
    (2, 0.7, 0.6, "Your order is on the way."),
```

## Priority queue

```
(1, 0.85, 0.3, "Refunds are processed within 5–7 business days.")
]

# Create a priority queue
pq = PriorityQueue()

# Insert responses with composite priority tuple
for urgency, purchase_score, browse_score, text in responses:
    # Lower urgency = higher priority (invert for max-heap behavior)
    priority = (-urgency, -purchase_score, -browse_score)
    pq.put((priority, text))

# Get the top-ranked response
top_response = pq.get()[1]

print("Customer Query:", customer_query)
print("Best Response:", top_response)
```

Got it — here are **coding-style questions with answers** based on your `PriorityQueue` example.

**Q1. Write a Python program** using `PriorityQueue` to store and retrieve tasks based on urgency level, where urgency 1 is the highest priority.

**Answer:**

```
from queue import PriorityQueue

tasks = PriorityQueue()
tasks.put((1, "Fix server issue"))
tasks.put((3, "Write documentation"))
tasks.put((2, "Update software"))

while not tasks.empty():
    print(tasks.get()[1])
```

**Output:**

```
Fix server issue
Update software
Write documentation
```

**Q2. Modify the program** so that the task with the **highest number** gets executed first (simulate max-heap).

**Answer:**

```
from queue import PriorityQueue
```

Priority queue

```
tasks = PriorityQueue()
tasks.put((-1, "Fix server issue"))
tasks.put((-3, "Write documentation"))
tasks.put((-2, "Update software"))
```

```
while not tasks.empty():
    print(tasks.get()[1])
```

**Output:**

```
Write documentation
Update software
Fix server issue
```

**Q3. Write code** to remove only tasks that have urgency less than 3, and display them.

**Answer:**

```
from queue import PriorityQueue
```

```
tasks = PriorityQueue()
data = [(1, "Fix server issue"), (3, "Write documentation"),
        (2, "Update software")]
```

```
for task in data:
    tasks.put(task)
```

```
removed = []
remaining = []
```

```
while not tasks.empty():
    task = tasks.get()
    if task[0] < 3:
        removed.append(task[1])
    else:
        remaining.append(task[1])
```

```
print("Removed:", removed)
print("Remaining:", remaining)
```

**Output:**

```
Removed: ['Fix server issue', 'Update software']
Remaining: ['Write documentation']
```

## Priority queue

**Q4. Write code** to store customer service responses with urgency and score, then display them in order of highest urgency and score.

**Answer:**

```
from queue import PriorityQueue

responses = [
    (1, 0.9, "Return policy"),
    (3, 0.5, "New arrivals"),
    (2, 0.8, "Order status"),
    (1, 0.85, "Refund time")
]

pq = PriorityQueue()

for urgency, score, text in responses:
    pq.put((-urgency, -score, text)) # Negative for max-heap

while not pq.empty():
    print(pq.get()[2])
```

**Output:**

```
Return policy
Refund time
Order status
New arrivals
```

**Q5. Write a Python program** to check if a `PriorityQueue` is empty before adding new items.

**Answer:**

```
from queue import PriorityQueue

pq = PriorityQueue()

if pq.empty():
    pq.put((1, "First task"))
    print("Added first task")

print("Is queue empty?", pq.empty())
```

**Output:**

```
Added first task
```

Priority queue

Is queue empty? False

I can also prepare **10 more small coding questions** that gradually increase difficulty if you want to practice more deeply. That way you'll be ready for both simple and tricky cases.