

Lab 4: Abstract Data Type Solution

2) Queues

1. Implement *Queue* class found in *myqueue.py*

```
1  ...
2  class Queue:
3      """A first-in-first-out (FIFO) queue of items.
4
5      Stores data in a first-in, first-out order. When removing an
6      item from the
7      queue, the least recently-added item (i.e. the oldest item in
8      the Queue)
9      is the one that is removed.
10     # === Private Attributes ===
11     # _items:
12     #     The items stored in this queue. The front of the list
13     represents
14     #     the front of the queue.
15     """
16     _items: List
17     def __init__(self) -> None:
18         """Initialize a new empty queue."""
19         self._items = []
20
21     def is_empty(self) -> bool:
22         """Return whether this queue contains no items.
23
24         >>> q = Queue()
25         >>> q.is_empty()
26         True
27         >>> q.enqueue('hello')
28         >>> q.is_empty()
29         False
30         """
31         return self._items == []
32
33     def enqueue(self, item: Any) -> None:
34         """Add <item> to the back of this queue.
35         """
36         self._items.append(item)
```

```

35     def dequeue(self) -> Optional[Any]:
36         """Remove and return the item at the front of this queue.
37
38         Return None if this Queue is empty.
39         (We illustrate a different mechanism for handling an
40         erroneous case.)
41
42         >>> q = Queue()
43         >>> q.enqueue('hello')
44         >>> q.enqueue('goodbye')
45         >>> q.dequeue()
46         'hello'
47         """
48         if self.is_empty():
49             raise EmptyStackError
50         else:
51             return self._items.pop(0)
52     ...

```

Listing 1: task_2.q1_solution.py

2. Complete functions *product* and *product_star* in *myqueue.py*

Notes:

- 오늘 형모 마음도 날씨도 봄 처럼 따뜻해요.
- 여보, 형모 사랑하는 내 여보 하고 손잡고 같이 걸을 수 있게 해줘서 고마워요 :)
- 형모 마음이 설레요
- 히히히히히히히
- 형모야. 오늘 무너지지 않고, 설레이는 마음 갖고, 차분히
- 화이팅 :)