Finger Exercises Lecture 20

The questions below are due on Monday November 21, 2022; 03:00:00 PM.

1) Question 1 of 1

In this problem, you will implement two classes according to the specification below: one `Container` class and one `Queue` class (a subclass of `Container`).
Our `Container` class will initialize an empty list. The two methods we will have are to calculate the size of the list and to add an element. The second method will be inherited by the subclass. We now want to create a subclass so that we can add more functionality -- the ability to remove elements from the list. A `Queue` will add elements to the list in the same way, but will behave differently when removing an element.

A queue is a first in, first out data structure. Think of a store checkout queue. The customer who has been in the line the longest gets the next available cashier. When implementing your `Queue` class, you will have to think about which end of your list contains the element that has been in the list the longest. This is the element you will want to remove and return.

```
class Container(object):
    """
    A container object is a list and can store elements of any type
    """
    def __init__(self):
        """
        Initializes an empty list
        """
        self.myList = []

    def size(self):
        """
        Returns the length of the container list
        """
        # Your code here

    def add(self, elem):
        """
        Adds the elem to one end of the container list, keeping the end
        you add to consistent. Does not return anything
        """
        # Your code here

class Queue(Container):
    """
    A subclass of Container. Has an additional method to remove elements.
    """
    def remove(self):
        """
        The oldest element in the container list is removed
        Returns the element removed or None if the stack contains no elements
        """
        # Your code here
```

```
# your class here
```

You have infinitely many submissions remaining.

Here is the solution we wrote:

```
class Container(object):
    def __init__(self):
        self.myList = []

    def size(self):
        return len(self.myList)

    def add(self, elem):
        self.myList.append(elem)

class Queue(Container):
    def remove(self):
        if self.size() > 0:
            return self.myList.pop(0)
        return None
```

MIT OpenCourseWare
https://ocw.mit.edu

6.100L Introduction to CS and Programming Using Python
Fall 2022

For information about citing these materials or our Terms of Use, visit: https://ocw.mit.edu/terms