Workshop #6: Lists

GitHub: abhijit-baruah

Implement the following for a Queue data structure:

- `__len__`
- `__bool__`
- `__repr__` ("unambiguous representation of an object")
- `__str__`
- `__contains__`

In [1]:

```
class Node:
    def __init__(self, init_data):
        self.data = init_data
        self.next = None

    def get_data(self):
        return self.data

    def get_next(self):
        return self.next

    def set_data(self, new_data):
        self.data = new_data

    def set_next(self, new_next):
        self.next = new_next
```

In [2]:

```
class UnorderedList:
    def __init__(self):
        self.head = None

    def add(self, item):
        temp = Node(item)
        temp.set_next(self.head)
        self.head = temp

    def length(self):
        current = self.head
        count = 0
        while current != None:
            count = count + 1
            current = current.get_next()
        return count

    def search(self, item):
        current = self.head
        found = False
        while current != None and not found:
            if current.get_data() == item:
                found = True
            else:
                current = current.get_next()
        return found

    def remove(self, item):
        current = self.head
        previous = None
        found = False
        while not found:
            if current.get_data() == item:
                found = True
            else:
                previous = current
                current = current.get_next()
        if previous == None:
            self.head = current.get_next()
        else:
            previous.set_next(current.get_next())

    def __len__(self):
        return self.length()

    def __bool__(self):
        return self.length() != 0

    def __repr__(self):
        return "UnorderedList()"

    def __str__(self):
        current = self.head
        v = "< UnorderedList: " + str(current.get_data())
        current = current.get_next()
        while current != None:
            v += ', ' + str(current.get_data())
            current = current.get_next()
        v += " >"
```

```
        return v

    def __contains__(self, item):
        return self.search(item)
```

In [3]:

```
mylist = UnorderedList()
```

In [4]:

```
mylist.add(31)
mylist.add(77)
mylist.add(17)
mylist.add(93)
mylist.add(26)
mylist.add(54)
```

In [5]:

```
# check __repr__
mylist
```

Out[5]:

```
UnorderedList()
```

In [6]:

```
# check __str__
print(mylist)
```

```
< UnorderedList: 54, 26, 93, 17, 77, 31 >
```

In [7]:

```
print(f'Length of the unordered list is: {len(mylist)}') # check __len__
```

```
Length of the unordered list is: 6
```

In [8]:

```
bool(mylist)
```

Out[8]:

```
True
```

In [9]:

```
emptylist = UnorderedList()
print(bool(emptylist))
```

```
False
```

In [10]:

```
print(f'Is 20 in the list? {20 in mylist}')
print(f'Is 77 in the list? {77 in mylist}')
```

```
Is 20 in the list? False
Is 77 in the list? True
```