Module 8 Homework - Sets

Create a set that provides O(1) running times for put and contains by using hashing. Do not use the python built-ins `set` or `dict`, but feel free to use the built-in `hash()` function to hash entries.

The starter code shows which methods you need to create, as well as doc strings defining what those methods should do.

Special Cases

- `__init__` has an optional `items` parameter. If a user specifies a collection during intialization, add all items from this collection to the set:

```
>>> s1 = MySet()
>>> print(s1)
MySet{}
>>> s2 = MySet([1, 2, 3])
>>> print(s2)
MySet(items={1, 2, 3})
```

- Maintain `O(n)` memory to store `n` items - this means you will need to rehash to a larger number of buckets when too many items are added, and rehash to a smaller number of buckets if too many items are removed.

Tests

Write unittests in a file `TestHw8.py`. Use test-driven development - write your tests first, then start coding in `hw8.py`.

Write one unittest per piece of functionality in the public interface. You don't need to explicitly test `_get_bucket()` or `_rehash`, since these are private attributes, but your test cases for adding/removing items should trigger rehashing events. That is, make sure you add enough items to force your set to rehash up, and then remove enough items to force your set to rehash down.

Test errors. You'll need to use a `with self.assertRaises()` statement to do this, see the unittest docs for an example (link)

Test a `True` and a `False` case for functions that return booleans (`contains` and `eq` in this assignment); e.g.:

```
x, y, z = 5, 5, 6
self.assertEqual(x ,y)
self.assertNotEqual(x, z)
```