Iterators

- `__iter__` returns the iterator object itself. This is used in `for` and `in` statements.
- `__next__` method returns the next value from the iterator.
- If there are no more items to return then it should raise an exception `StopIteration` exception.

```
class Counter(object):
    def __init__(self, lower, upper):
        self.current = lower
        self.high = high

    def __iter__(self):
        'Returns itself as an iterator object'
        return self

    def __next__(self):
        'Returns the next value till current is lower than high'
        if self.current > self.high:
            raise StopIteration
        else:
            self.current += 1
            return self.current - 1
```

```
>>> c = Counter(5,10)
>>> for i in c:
...   print(i, end=' ')
...
```

💡 Remember that an iterator object can be used only once. It means after it raises *`StopIteration`* once, it will keep raising the same exception.