L11 Problem 5

(5 points possible)

ESTIMATED TIME TO COMPLETE: 10 minutes

Consider the following code from the last lecture video:

```
class intSet(object):
    """An intSet is a set of integers
    The value is represented by a list of ints, self.vals.
    Each int in the set occurs in self.vals exactly once."""

    def __init__(self):
        """Create an empty set of integers"""
        self.vals = []

    def insert(self, e):
        """Assumes e is an integer and inserts e into self"""
        if not e in self.vals:
            self.vals.append(e)

    def member(self, e):
        """Assumes e is an integer
           Returns True if e is in self, and False otherwise"""
        return e in self.vals

    def remove(self, e):
        """Assumes e is an integer and removes e from self
           Raises ValueError if e is not in self"""
        try:
            self.vals.remove(e)
        except:
            raise ValueError(str(e) + ' not found')

    def __str__(self):
        """Returns a string representation of self"""
        self.vals.sort()
        return '{' + ','.join([str(e) for e in self.vals]) + '}'
```

Your task is to define the following two methods for the `intSet` class:

1. Define an `intersect` method that returns a new `intSet` containing elements that appear in both sets. In other words,

   ```
   s1.intersect(s2)
   ```

   would return a new `intSet` of integers that appear in both `s1` and `s2`. Think carefully - what should happen if `s1` and `s2` have no elements in common?

2. Add the appropriate method(s) so that `len(s)` returns the number of elements in `s`.

   Hint: look through the Python docs to figure out what you'll need to solve this problem.

```
class intSet(object):
  """An intSet is a set of integers
  The value is represented by a list of ints, self.vals.
  Each int in the set occurs in self.vals exactly once."""

  def __init__(self):
    """Create an empty set of integers"""
    self.vals = []

  def insert(self, e):
    """Assumes e is an integer and inserts e into self"""
    if not e in self.vals:
      self.vals.append(e)

  def member(self, e):
    """Assumes e is an integer
```

Unanswered


Download on the
App Store
ANDROID APP ON
Google play