CS417 Programming Assignment 6

Due: Friday, March 27th. Late penalty: Sat/Sun/Mon 5%, Tue 10%, Wed 20%, Thu 50%, Fri 100%.

Overview

This assignment focuses on classes and objects. You should implement a container class. It works as a set, because each value is stored only once: it does not hold duplicate values.

Begin by creating a directory for your work, and download the starting file into it: `a_set.py` .

The class `a_set` uses a simple list to store the values. You MAY NOT use python's built-in `set` class in your implementation. Since python's set class uses a hash table, your implementation will be slower for some operations.

Your Tasks

I have implemented the first three methods, and you must implement the others. Here are the class methods:

Method	**Description**	**Done?**
`__init__`	Constructor.	Yes
`copy`	Make a copy of this set	Yes
`__iter__`	Generator which visits all the values	Yes
`add(value)`	Add value to the set, unless it's already in the set	NO
`__contains__(value)`	Return True / False if value is / isn't in the set	NO
`remove(value)`	Remove value from the set	NO
`union(other)`	Create and return new `a_set`, which holds all the values that occur in `self` *or* in `other`	NO
`intersection(other)`	Create and return new `a_set`, which holds only the values that occur both in `self` *and* in `other`	NO
`difference(other)`	Create and return a new a_set, which holds all the values that occur in `self` but do *not* occur in `other`	NO
`issubset(other)`	Return True / False if self is / isn't a subset of `other`. A is a subset of B if every element in A also occurs in B.	NO
`len`	Return the number of values	NO
`__repr__`	Return a programmer-friendly string that describes self. It should mention that it's a	NO

	a_set, and should have all its values.	
__str__	Return a user-friendly string that describes the contents of self. It should begin with '{' and end with '}'. Each value should be followed by a comma and a space, except the last value.	NO

Note that the constructor has an optional argument orig , which defaults to None if omitted. If orig is omitted, create an empty set. If orig is another a_set, make a copy of it (like a copy constructor in C++). If it is a list, create a set with that list's elements.

Test Your Code

The starter file has a main function that tests the class, but does not check all the cases. Expand it. Here is some output when I test my code:

```
Empty set: {}
Testing "add()":
Testing "__str__":
  A = {3, 1, 4, 5, 9, 2, 6}
  B = {8, 9, 7, 3, 2, 4, 6}
Testing "__repr__":
  repr(A) = a_set(3, 1, 4, 5, 9, 2, 6)
  repr(B) = a_set(8, 9, 7, 3, 2, 4, 6)
Testing "remove":
  After deletions:
  A = {1, 4, 5, 2, 6}
  B = {9, 3, 2, 4, 6}
  CORRECT: Exception raised when deleting absent key
Testing "contains":
           0   1   2   3   4   5   6   7   8   9
  in A?    .   Y   Y   .   Y   Y   Y   .   .   .
  in B?    .   .   Y   Y   Y   .   Y   .   .   Y
Testing "union":
  A union B = {1, 4, 5, 2, 6, 9, 3}
Testing "intersection":
  A intersect B = {4, 2, 6}
Testing "difference":
  A minus B = {1, 5}
  B minus A = {9, 3}
Testing "issubset":
  A subset of B? False
  B subset of A? False
  A subset of A? True
Testing "__len__":
  len(A) = 5
  len(B) = 5
```

In case you are curious, there is a good description of all of python's "magic" methods, such as __iter__. It was written by Rafe Kettler (although his website expired, and rafekettler.com now hosts malware!). I'm including a .pdf copy of Rafe's work.

Submitting your work

When you finish, go to `mycourses.unh.edu`, find cs417 and assignment 6, and click the "Submit" button. Then upload `a_set.py` .