CSC 242

Exercise 1

A simple software system for a library models a library as a collection of books and patrons. A patron can have at most three books out on loan at any given time. Each book has a title, an author, a patron to whom it has been checked out, and a list of patrons waiting to borrow it. Each patron has a name, the number of books it has currently checked out, and a list of the books currently checked out. Develop the classes **Book** and **Patron** to model these objects. Test these classes in a driver program.

Here is the starter code for the two classes:

```
from collections import deque

class Book(object):
    '''
    This class will represent all kinds of Books at a library.
    '''
    def getBookOwner(self):
        return self._patronOwner
    def queueLength(self):
        return len(self._patronQueue)
    def getPatronQueue(self):
        return self._patronQueue

    #Pre-condition: book and patron exists
    #Post-condition: 1. patron borrows the book if not currently out on loan and
    #                   nobody is waiting
    #                   or if not currently out on loan and patron is next in the
    #                   queue; add the book to the patron's list of books;
    #                   return True
    #                2. patron is added to the book's queue; return False
    #                3. patron isn't allowed to borrow more than max books (3);
    #                   return False
    def borrow(self, patron):
        '''Finish'''

    #Pre-condition: book and patron exists
    #Post-condition: 1. book is removed from patron's list of books,
    #                   if book is in care of patron; current owner of the book is
    #                   set to None
    #                2. book is not returned because patron doesn't have it in the
    #                   first place
    def returnBook(self, patron):
        '''Finish'''

    def __init__(self, title, author):
        '''
        Constructor
        '''
        self._patronQueue = deque() #use append and popleft for queue operations
        self._title = title
        self._author = author
        self._patronOwner = None
```

```
    def __str__(self):
        if self._patronOwner != None:
            s = self._title + ", " + self._author + " in care of: " + \
                str(self._patronOwner)
        else:
            s = self._title + ", " + self._author + " and has not been borrowed.\n"

        s += "Waiting:\n"
        count = 1
        for item in self._patronQueue:
            s += str(count) + ". " + str(item)
            count += 1
        s += "\n"
        return s


class Patron(object):
    '''
    This class will represent all kinds of Books at a library.
    '''
    maxBooks = 3

    #Pre-condition: book exists
    #Post-condition: 1. book is removed from the patron's list of books;
    #               the number of books the patron has checked out is decremented by 1
    #                 2. a message is displayed stating the patron does not have the
    #                    book checked out
    def removeBook(self, book):
        '''Finish'''

    #Pre-condition: book exists
    #Post-condition: 1. book is added the patron's list of books,
    #                   as long as the patron has less than 3 books checked out;
    #                   the number of books the patron has checked out is incremented
    #                   by 1; return True
    #                 2. a message is displayed stating the patron has reached their
    #                    max and can't borrow anymore books; return False
    def addBook(self, book):
        '''Finish'''

    def __init__(self, name):
        '''
        Constructor
        '''
        self._name = name
        self._numBooks = 0
        self._books = []

    def __str__(self):
        s = self._name + " has " + str(self._numBooks) + " books.\n"
        return s

    def getBooks(self):
        return self._books
```

Here is a sample driver program:

```
from book import Book
from patron import Patron

def main():
    book1 = Book("Of Mice and Men", "Steinbeck")
    book2 = Book("The Great Gatsby", "Fitzgerald")
    book3 = Book("1984", "Orwell")
    book4 = Book("One Flew Over the Cuckoo's Nest", "Kesey")
    patron1 = Patron("Ivan")
    patron2 = Patron("Jimmy")
    patron3 = Patron("Bob")

    print("Book1: " + str(book1))
    print("Patron1: " + str(patron1))

    book1.borrow(patron1)  #borrow calls patron.addBook
    book1.borrow(patron2)
    book1.borrow(patron3)

    book2.borrow(patron1)

    book3.borrow(patron1)

    #patron1 should not be able to borrow over the max limit (3 books)
    book4.borrow(patron1)

    book4.borrow(patron2)

    print("Book1: " + str(book1))
    print("Patron1: " + str(patron1))

    book1.returnBook(patron1)
    print("Book1: " + str(book1))

    #Try to borrow Book1 to Bob.
    book1.borrow(patron3)
    #Try to borrow Book1 to Jimmy.
    book1.borrow(patron2)

    print("Book1: " + str(book1))
    print("Patron2: " + str(patron1))

if __name__ == '__main__':
    main()
```

Here is the associated output:

```
Book1: Of Mice and Men, Steinbeck and has not been borrowed.
Waiting:

Patron1: Ivan has 0 books.

Book is available. Borrow to: Ivan
Book is not available. Add: Jimmy to the queue.
Book is not available. Add: Bob to the queue.
Book is available. Borrow to: Ivan
Book is available. Borrow to: Ivan
Can't borrow more books--MAX REACHED!
Book is available. Borrow to: Jimmy
Book1: Of Mice and Men, Steinbeck in care of: Ivan has 3 books.
Waiting:
1. Jimmy has 1 books.
2. Bob has 0 books.

Patron1: Ivan has 3 books.

Returned: Of Mice and Men, Steinbeck in care of: Ivan has 2 books.
Waiting:
1. Jimmy has 1 books.
2. Bob has 0 books.

Book1: Of Mice and Men, Steinbeck and has not been borrowed.
Waiting:
1. Jimmy has 1 books.
2. Bob has 0 books.

Bob is not next in the queue to borrow:
Of Mice and Men, Steinbeck and has not been borrowed.
Waiting:
1. Jimmy has 1 books.
2. Bob has 0 books.

Book is available. Borrow to: Jimmy
Book1: Of Mice and Men, Steinbeck in care of: Jimmy has 2 books.
Waiting:
1. Bob has 0 books.

Patron2: Ivan has 2 books.
```