Term Project
CS 2210

Due: 5pm

Develop a Java package which implements an ADT for a (2,4) search tree, as described on page 500 of the text. You will work in teams of two.

Guidance:

a. Your ADT should be able to store ordered information, and be accessed according to the Dictionary interface (i.e., it should implement the dictionary interface). At instantiation, the user will pass to your constructor a reference to a Comparator (a class which implements the Comparator interface) which will be used to order the data in the tree by keys. In addition to the functionality provided by the Dictionary interface, your ADT should provide a printAllElements() method which writes the current Dictionary status to the console.
b. You should design your ADT such that is can store any Object. However, you only need to demonstrate that it works properly for the Integer type. I will provide an IntegerComparator you can use. If the user provides a key that is not of the expected type (in our case Integer), you should throw an InvalidObjectException. I will also provide draft code for the required printAllElements() method.
c. Your ADT should be thoroughly tested. You should develop and turn in test code which instantiates the ADT, performs Dictionary operations on it, and then does printAllElements() to verify that the data is being stored properly. You only need demonstrate (via turn-in) that you have tested the ADT at the interface level; i.e., you only need to instantiate the ADT and test it via the Dictionary operations.
d. You are NOT required to use JUnit; testing in main is fine. Testing should include an automated test that inserts and removes many (> 1000) random integers. Removal should fully empty the tree.
e. I will provide the Dictionary interface. Note this is subset of the functions the text describes. You are only required to provide the functions I specified in the provide interface.
f. Your code should be well commented as we have been doing all semester. In particular, the code which manipulates the (2,4) search tree is quite complicated and should be well-explained by comments. You are not responsible for commenting the code I provide you.
g. I strongly recommend you use Netbeans. Without its debugger, this project will be extremely tough.
h. Big hint: getting pointers hooked up right is the biggest complication. I have provided you a *checkTree*() method of the TwoFourTree that will check that parent and child pointers are bi-directional. It can save you lots of grief.

Term Project
CS 2210 – Spring 2016

Required for turn-in:

a. Listings of all code, including test code (properly commented, and in compliance with CS Java style guide).
b. CS cover sheet, including a signed statement regarding how well your final code worked.
c. A demo of the completed project will be required.