BoxesManager is data structures based exercise. The whole structure is composed of BST(Binary Search Tree) and LinkedList with few modification.

For the BST I add two extension methods:

- SearchBiggerOrEquelTo – get a value “x” and return true if there is an equal or bigger value than x. if so, the method pop out the smallest (or equal) value bigger than x.
- SearchBiggerThan – get a value “x” and return true if there is a bigger (and not equal) value than x. is so, the method pop out the smallest value bigger than x.

For the LinkedList I modify one extra method (not trivial for any LinkedList):

- ModifyNode – “pull out” a specific node from the list, make a manipulation on the data according to given delegate and put back the node in the end of the list.

Using those data structures I wrote the BoxesManger Class. The logic behind this class is that every given box data will split into two BST, main BST and “nested” BST – the x value will be saved on a main tree and for each x value there a link to another BST that saves the y values of that x.

In addition, the nested BST data will save the amount of boxes and a reference to a linked list node. The purpose of the linked list is to save the boxes data chronologically ordered by the time of using – so we could find efficiently the unused boxes. (GetRidOfBoxes()). In addition, the data of the linked list include references to both main tree and height tree data. (so when we removing unused boxes, the application will find the suitable nodes in the trees and will remove those nodes as well)