You will submit your solution to this assignment to the Curator System (as `HW1`). Your solution must be either a plain text file (e.g., NotePad) or a typed MS Word document; submissions in other formats will not be graded.

Partial credit will only be given if you show relevant work.

For questions 1 and 2, refer to the BST generic interface provided for Minor Project 2. Strive for the most efficient solution you can devise. You may not use any Java collections, including ones you implement yourself, to store any additional information (like copying the data elements from the BST into an array).

1. [25 points] Design an algorithm to determine whether a given binary tree is *full*, as defined in the course notes. Express your solution using Java code as if it were to be implemented as a member function the BST generic specified in Minor Project 2; your answer must conform to the public interface shown below:

```
// Pre:      none
// Post:     the BST is unchanged
// Returns:  true iff the BST is full
//
public boolean isFullBinaryTree( ) {

   // up to you. . .
}
```

It is perfectly acceptable to make use of a private helper function, in which case you must show implementations of both functions.

2. [25 points] Design an algorithm to determine whether a BST contains any duplicate values; that is, answer the question: is there any value **X** such that **X** occurs in two different nodes within the tree. Assume that duplicates of an existing value will have been inserted to the right of the original copy.

Write an implementation of your algorithm, using Java syntax, as if the implementation were to be placed within the BST implementation from Minor Project 2. Your answer must conform to the public interface shown below:

```
// Pre:      none
// Post:     the BST is unchanged
// Returns:  true iff two different nodes in the BST contain values for
//           which equals() returns true
//
public boolean hasDuplicates( ) {

   // up to you. . .
}
```

It is perfectly acceptable to make use of a private helper function, in which case you must show implementations of both functions.

3. A simple, unbucketed PR quadtree is used to organize data object that lie within a square bounded by the corners (0, 0) and (1024, 1024). Currently, the tree contains a single data object, which lies at the coordinates (25, 25). The following questions are independent.

a) [5 points] Give specific coordinates such that the insertion of a second data object at those coordinates will cause exactly one splitting operation to occur.
b) [10 points] Give specific coordinates such that the insertion of a second data object at those coordinates will cause exactly three splitting operation to occur.

c) [10 points] Give specific coordinates such that the insertion of a second data object at those coordinates will cause exactly six splitting operation to occur.

4. [25 points] A simple, unbucketed PR quadtree is used to organize data object that lie within a square bounded by the corners (0, 0) and (1024, 1024). When the first two data objects are inserted, one splitting operation occurs. Neither of those data points lies on the boundary between two regions. What is the minimum distance the two data objects could be separated by? Why? Explain clearly.