PROBLEM SOLUTION APPROACH

Q1 1= I HAVE USED RECURSİVE METHOD.IT'S PRETTY SIMPLE ONLY A FEW LINES OF CODE

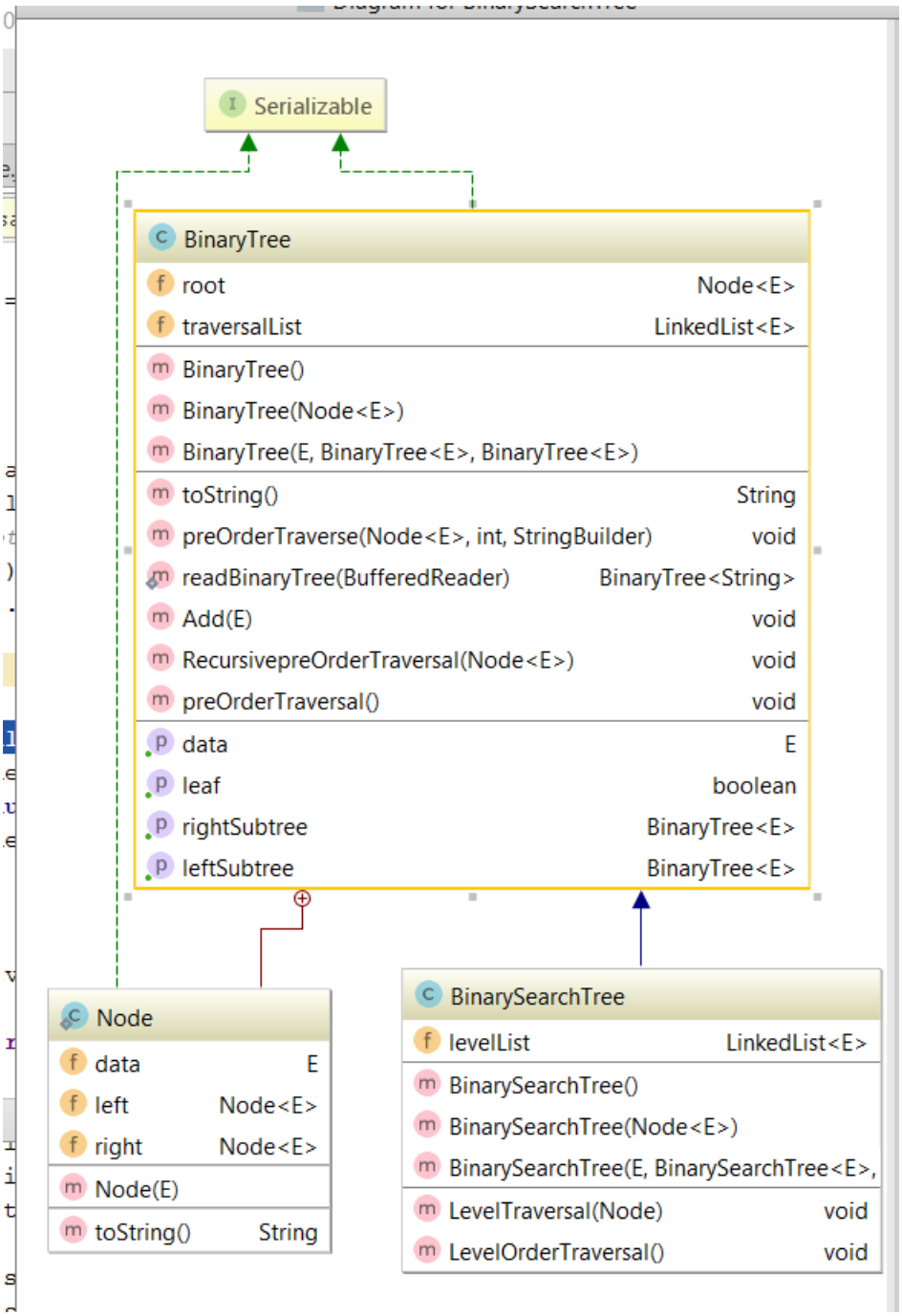
I HAVE ADDED THE DATA TO A LINKEDLIST AS I READ THEM.I HAVE KEPT THE LIST AS A CLASS MEMBER.THEN USE AN ITERATOR TO MOVE ON THE LIST AND PRINT THE LIST.

Q1 2= WE USE QUEUE METHODS.WE FILL THE QUEUE BOTH LEFT AND RİGHT NODE AND WE POLL() THE FIRST ELEMENT WHICH IS LEFT.BUT THE RIGHT ELEMENT IS STILL THERE.THEN WE PROCEED TO ADD BOTH LEFT AND RIGHT NODE OF THAT POLLED NODE TO THE QUEUE.THEN WE POLL THE FIRST RIGHT NODE WE ADDED AND ADD IT'S LEFT AND RIGHT NODE IN THAT ORDER TO QUEUE.

IN TIME EACH ONE WILL GET IT'S TURN.AND ALL OF THOSE POLLED NODES DATA'S ADDED TO A LINKED LIST WHICH WE WILL BE USING AN ITERATOR TO MOVE ON THE LIST AND PRINT THE LIST.

Q2 = I HAVE AN `RecursiveTraversal` **METHOD WHICH TRAVERSES ON A BINARY TREE AND ADDS THEM IN AN LINKEDLIST WHICH I KEEP AS A MEMBER VARIABLE.**
THEN I USE THIS LIST TO FIND THE PARENT OR GRANDPARENT,DEPENDS ON EBU,IBN STUATION.THEN I PROCEED THE CHECK THE CHILDS.

CLASS DIAGRAMS



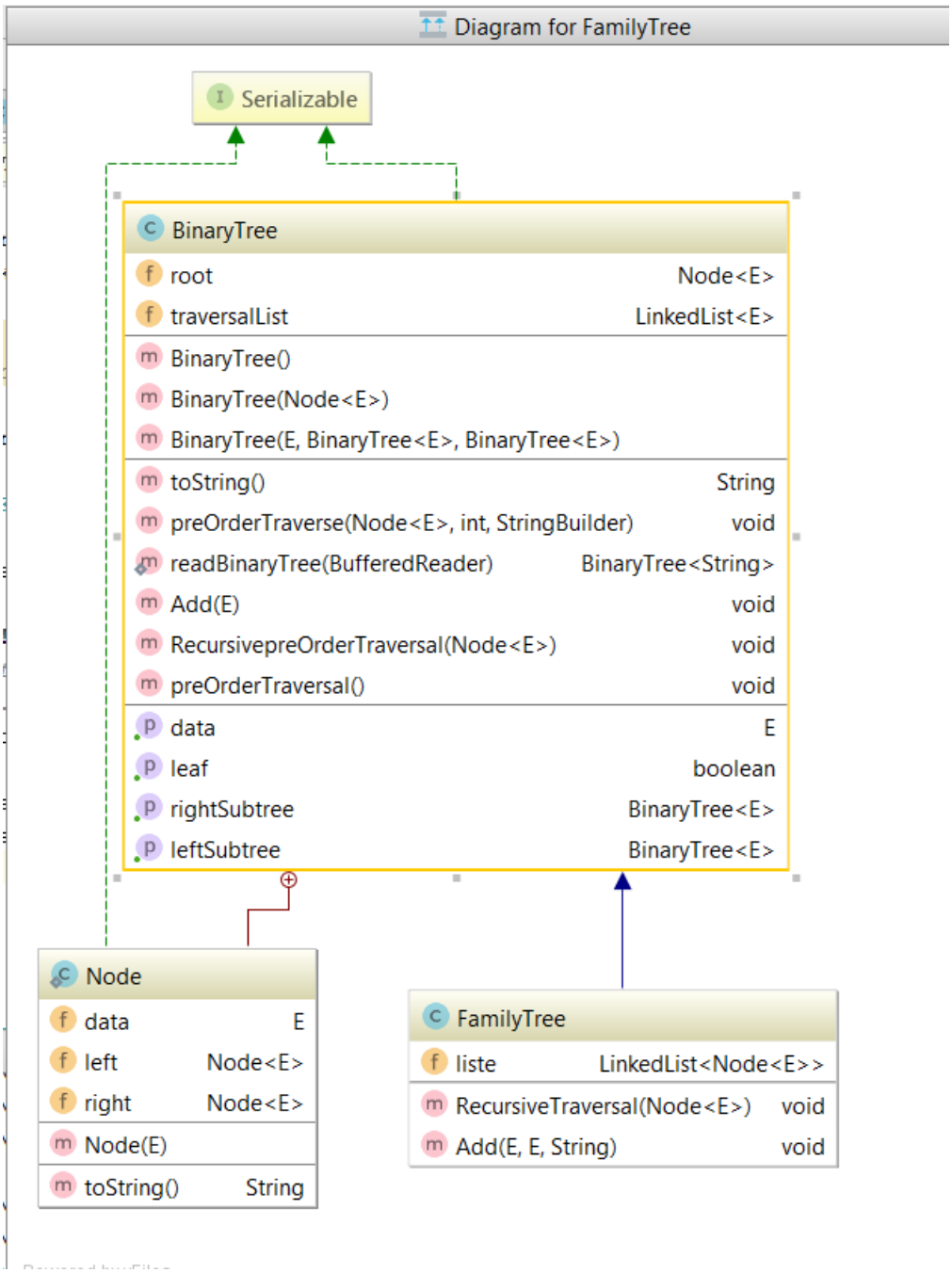


Diagram for FamilyTree
Serializable
BinaryTree
root Node<E>
traversalList LinkedList<E>
BinaryTree()
BinaryTree(Node<E>)
BinaryTree(E, BinaryTree<E>, BinaryTree<E>)
toString() String
preOrderTraverse(Node<E>, int, StringBuilder) void
readBinaryTree(BufferedReader) BinaryTree<String>
Add(E) void
RecursivepreOrderTraversal(Node<E>) void
preOrderTraversal() void
data E
leaf boolean
rightSubtree BinaryTree<E>
leftSubtree BinaryTree<E>
Node
data E
left Node<E>
right Node<E>
Node(E)
toString() String
FamilyTree
liste LinkedList<Node<E>>
RecursiveTraversal(Node<E>) void
Add(E, E, String) void