Introduction:

In this part we will work together to build all potential paths from starting point [0, 0] leading to the GOAL using tree representing in the chessboard. Each node in the tree corresponds to a step in the path (a ChessboardPosition). Path forks are represented as branches to different children, corresponding to the alternative steps one could follow from a certain position.

Beware that this strategy of building all potential paths grows exponentially and leads easily to combinatorial explosion. This strategy can only be applied to small chessboards, or more accurately, to chessboards with few degrees of freedom for choosing the path. So in this part of the project we have updated ***ChessboardSamples*** class to minimize the potential paths.

Task #4a: Explore all children

In **PathsTreeNode** class (in PathsTree.java file) complete **exploreAllChildren()** method to explore all the children of a PathsTreeNode object. *(More details commented in that method)*

So by the end you will get a tree of all possible paths even if some of the paths don't lead you to a GOAL.

Task #4b: Finding all paths

In **PathsTree** class complete **findAllPaths()** method to return all the valid paths as a DLLPath objects in the ArrayList. A valid path is a path leading to GOAL position. *(More details commented in that method)*

Notes:

- You shouldn't use any print instruction in your solution.

- Make sure your submission does not contain any copied (or very similar) code from any source. Regarding that the lab exam committee will run a script of plagiarism check and if your submission contains any copied (or very similar) code you will get 0 of 10 for the project grade.