# Folio Task Reflection

This algorithm runs through all possible placements of the princess and determines if it is valid. If it isn’t valid, the algorithm backtracks to try a different option. The algorithm has a time complexity of at least O(n!).

## Challenges:

I found this task very tricky to fully complete. While the solution provided solves to some of the solutions, it does not provide all solutions. The algorithm does not get solutions for configurations where pieces fall on the same column.

Tasking the program to check whether each piece was attacking a square without going outside the bounds of the matrix was another problem. This was fixed by adding if statements to break the loop when it goes over.

## Alternatives:

An iterative approach was tried before the recursive approach. Using a combination of nested loops every spot would be tried and a piece would be placed till all pieces were place. While this provided a solution, it did not provide all solutions, so it was abandoned.

A brute force approach would have involved generating all possible placements of the Princess piece and selecting the ones that fulfilled the two square distance. I chose to not peruse this since it would have a larger time complexity and an even larger space complexity for storing all the board configuations.