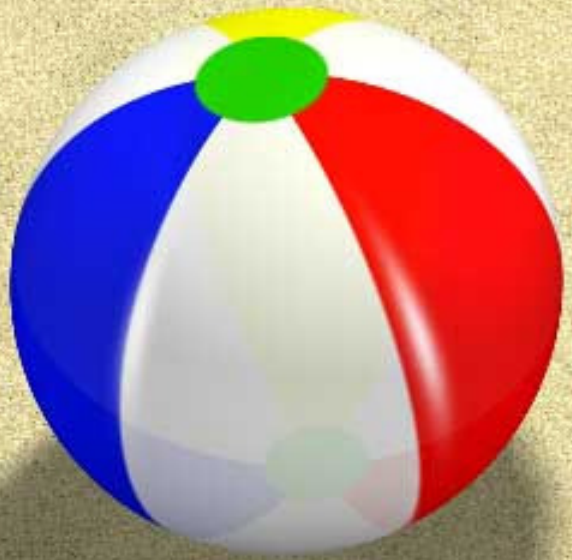
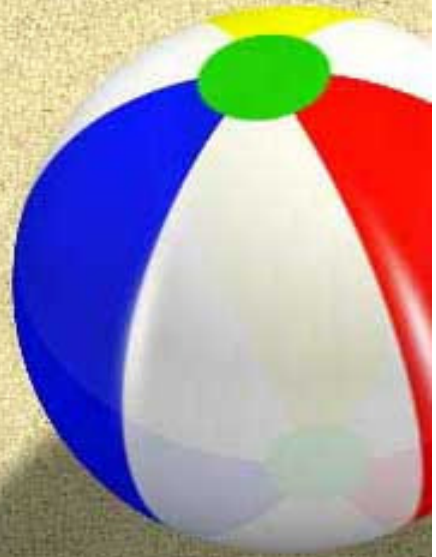
A Knight and A Queen



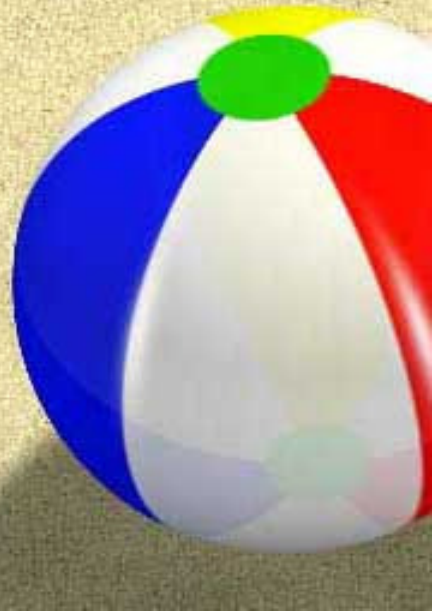
Problem

- Given a Knight and a Queen on a chessboard, find if the Knight can reach the Queen within a bounded number of moves k



First Idea

- Breadth First Search (BFS)

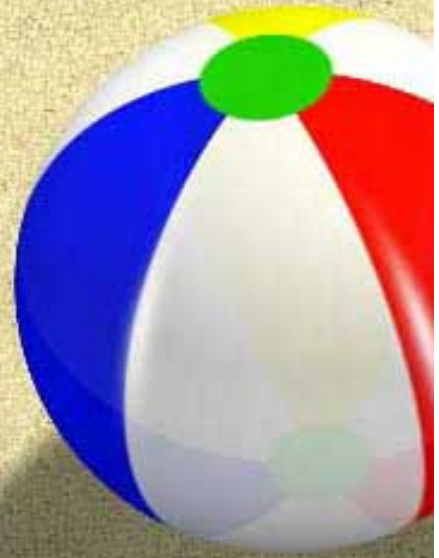


Positions and Levels

- Represent positions on the board as triples (x, y, l)
- (x, y) represents the coordinates of the position on the board
- l represents the minimum number of moves from the starting position to reach the current position

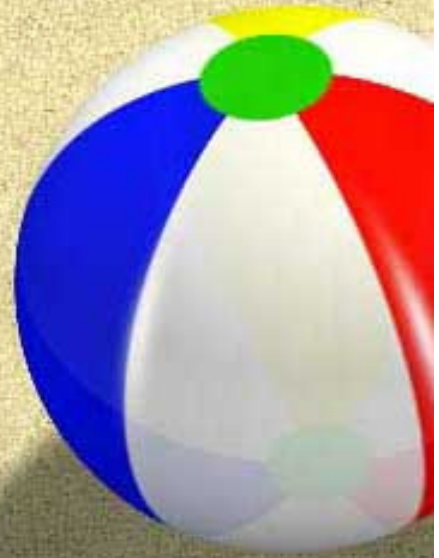
Algorithm

- Put the position of the Knight in a queue as well as in a set S with level 0
- While the queue is not empty
  - Pop a position p from the front of the queue. If its level is k ignore it and continue to the next iteration.
  - Consider the 8 neighbor positions the Knight can go to from p
  - Add those that are on the board and not in S to the queue with level one more than the level of p



Algorithm

- At the end, S contains all positions reachable from the Knight's start position within k moves
- Check if Queen's position is in the set S (note that the set ignores levels of positions)

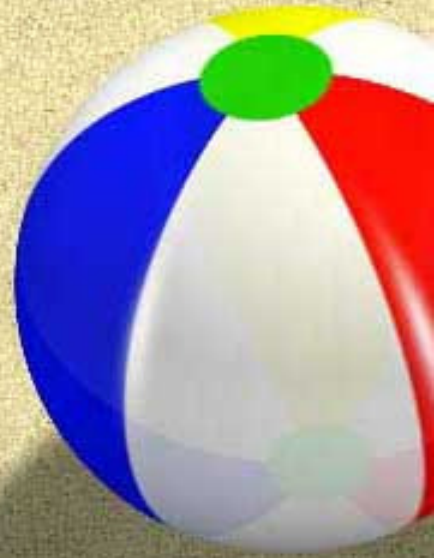


Input Size

- Pay attention to the bounds of the size of the input in the problem statement
- The board can be as big as 1000000 x 1000000
- You cannot use a matrix to keep track of visited positions during the BFS

Input Size

- The number of moves can be as big as 256
- How large can S get? A position can have up to 8 children!
- Try it! Print the size of S at the end of the algorithm when the start position is sufficiently far from the edges of the board
- `915973!`
- `Times out!`



A Better Idea

- Generate reachable positions from both Knight's and Queen's start positions going as long as $\frac{k}{2}$ moves into sets S_1 and S_2
- Check if their intersection is nonempty. If yes, the Queen is reachable by the night within k moves
- When $k = 256$, S_1 and S_2 have at most `228613` elements. Better!